DTIC FILE COPY

(4)

AD-A204 410

Contract N00014-87-C-2547

THEORETICAL RADIATION PATTERNS OF NAVSPASUR TRANSMITTER ANTENNAS

DTIC
ELECTE
FEB 0 2 1989
S D
D

INTERFEROMETRICS INC.

Dr. Steven L. Berg

30 November 1988

89 1 3 145

EXECUTIVE SUMMARY

Interferometrics Inc. has completed a series of radiation pattern calculations for each of the NAVSPASUR transmitter sites. As an integral part of this effort we have performed the first detailed calculation of the radiation pattern of the NAVSPASUR transmitter element over a finite-sized wire-grid ground screen.

The transmitter antenna pattern parameters which we compute are consistent with the reported NAVSPASUR values in the far field and provide a meaningful calculation of the antenna near-field performance. These results can be used as an integral component of any NAVSPASUR model.

The tools developed to generate these patterns can be used for a number of other related applications, and therefore provide a strong foundation for future NAVSPASUR studies.

CONTENTS

FIGURES

I. INTRODUCTION

Interferometrics Inc. has recently participated in an effort to determine the capabilities of the current Naval Space Surveillance System (NAVSPASUR), and to examine the impact of proposed system enhancements. As part of this effort, Interferometrics Inc. has identified the need for an accurate analytic model of the NAVSPASUR system which can be used to simulate data. One of the critical components of such a model is a realistic transmitter antenna radiation (or beam) pattern. The transmitter beam pattern calculations previously available have been based on oversimplified models of the dipole element radiation pattern. We provide here a detailed calculation of the radiation pattern of the NAVSPASUR transmitter element above its finite wire-grid ground screen. Using this calculation, we also derive near-field and far-field beam patterns for the NAVSPASUR transmitter antennas for a number of interesting cases.

A detailed description of the transmitter antennas is given in Section II and an overview of the terminology used in this paper is contained in Section III. In Section IV we calculate the beam patterns for a single NAVSPASUR radiating element in free space, over an infinite ground screen, and over a finite ground screen. The finite ground-screen results are used in Section V to calculate the antenna array radiation patterns using the principle of superposition. We present in Section VI our conclusions, together with a discussion of the potential applications of the model we have developed. An overview of the computer codes developed for these and other related calculations is given in the Appendix to this report.

II. ANTENNAS

A. Configuration

The present NAVSPASUR transmitting system consists of three separate transmitters positioned on a great circle across the southern United States. The transmitting antenna at each site consists of a linear array of dipole elements aligned in a north-south direction. Each site transmits an unmodulated continuous-wave signal at a frequency (f) of 216.980 MHz, corresponding to a wavelength (λ) of 1.38 meters.

The 810 kW transmitter at Lake Kickapoo, Texas is the most powerful and the longest, consisting of eighteen separate collinear bays stretching 3200 m in the north-south direction. Each bay contains 144 elements spaced 1.27 m (0.92 λ) apart, except for bay #8 (numbering from north to south), which is split by a road and consists of two half bays with 54 elements each. The end elements of adjacent bays are separated by 3.81 m. The distance between the elements at the road gap is 73.2 m.

The Kickapoo transmitter is referred to as the Kickapoo complex, since it is created from two smaller nine-bay transmitters called North Kickapoo and South Kickapoo. Each half can be operated as an individual transmitter antenna.

The Gila River, Arizona and the Jordan Lake, Alabama transmitters each supply 45 kW of power to single-bay antenna arrays. The Gila River transmitter has 384 elements spaced 1.30 m (0.94 λ) apart, while the Jordan Lake transmitter has 256 elements spaced 1.22 m (0.88 λ) apart.

B. Element

The NAVSPASUR transmitter radiating element is a half-wave, center-fed aluminum dipole with an arrowhead configuration. Each of the two quarter-wave monopoles, which together make up the dipole, is fixed at a 55° droop angle (measured with respect to the horizontal), as shown in Fig. 1. The center of the element is positioned 0.517 m (0.375 λ) over the ground screen. The element is parallel to the great circle plane.

C. Ground Screen

The transmitter antenna ground screens are a wire grid 3.66 m (2.66 λ) wide. The lengths of the ground screens are identical to the transmitter antenna lengths of 500.0 m and 314.6 m for the Gila River and Jordan Lake antennas, respectively. For the Kickapoo complex, the ground screen for each bay is equal to the bay length, which is about 183 m for all bays except #8. The two halves of bay #8 have ground screens with lengths about 68.6 m each.

The existing ground screen is made from commercial wire fencing of unknown composition. The ground screen grid wires are spaced 0.076 m (0.055 λ) apart running parallel to the element plane, with thirteen plane-perpendicular wires, spaced .305 m apart, running the length of the ground screen. The wire diameter is 0.0033 m (0.002 λ).

The ground screens are scheduled to be replaced in 1989. The replacement ground screen has bars instead of wires with closer spacings of 0.051 m parallel to the element plane. The rectangular cross-sectional bars are made of an aluminum alloy, with a width of 0.0254 m and a thickness of 0.0032 m.

III. TERMINOLOGY

A number of terms used herein may require either definition or clarification. This section presents an overview of the terminology as applied in the context of this paper.

The radiation or beam pattern refers to the angular dependence of the radiation properties of an antenna. A beam pattern is generally presented as a plot of some parameter proportional to radiation intensity versus angle, often in polar plot format. The intensity-proportional parameter may be gain, directivity, radiation intensity, or power density. Antenna refers to either a single radiative element or an array of elements.

The terms far (electric) field and near (electric) field are used throughout this paper. When a single element is of interest, the near field refers to the *reactive* near field. The

reactive near-field region is defined in terms of the complex Poynting vector, which is related to the electric field (**E**) and magnetic field (**H**) by the equation

$$\mathbf{S}(r,\theta,\phi) = \frac{1}{2}\,\mathbf{E}(r,\theta,\phi) \times \mathbf{H}^*(r,\theta,\phi)\,. \tag{1}$$

The reactive near-field region is defined as the range of distances for which terms in the Poynting vector proportional to $1/r^n$ with $n > 2$ are significant. These terms, which are imaginary, correspond to the reactive power, which represents stored energy. The term proportional to $1/r^2$ corresponds to the radiative power.

The near field of an array of elements refers to the ***radiative*** near field. The radiative near-field region (also referred to as the Fresnel region) lies between the reactive near-field region of the elements and the array far-field region. The array far-field region (also referred to as the Fraunhofer region) is defined as the range of distances where the approximation that the radiation from all points of the array travel parallel paths to the target is valid. It is generally defined by

$$r \geq \frac{2\mathrm{D}^2}{\lambda}\,, \tag{2}$$

where D is the largest linear dimension of the array. For the radiative near-field patterns, power density ($\mathcal{P}$) is a convenient plot variable, since the target cross section multiplied by power density yields the total power power incident on the target. It is defined as the magnitude of the real part ($\Re$) of the Poynting vector, or

$$\mathcal{P}(r,\theta,\phi) \equiv \frac{1}{2}\,\left|\Re\left[\mathbf{E}(r,\theta,\phi) \times \mathbf{H}^*(r,\theta,\phi)\right]\right|\,. \tag{3}$$

Directivity, as used in this paper, is defined by the equation

$$D(\theta,\phi) \equiv \frac{U(\theta,\phi)}{U_{av}}\,, \tag{4}$$

where $U(\theta,\phi)$ is the radiation intensity in the direction (θ,ϕ), given by

$$U(\theta,\phi) = r^2\,\mathcal{P}(r,\theta,\phi)\,, \tag{5}$$

and U_{av} is radiation intensity averaged over all direction angles. Gain is then defined as

$$G(\theta, \phi) \equiv \varepsilon D(\theta, \phi) , \tag{6}$$

where ε is the radiation efficiency, defined as

$$\varepsilon \equiv \frac{P_{rad}}{P_{in}} . \tag{7}$$

Here P_{rad} is the total radiated power and P_{in} is the input power to the antenna. Since the radiation efficiency of the NAVSPASUR's transmitter antennas is unknown, directivity will be used as the plot parameter for far-field beam patterns. Gain and directivity are also commonly used to represent the maximum values of Eqs. (4) and (6). If a single value is given as the gain or directivity of an antenna, the maximum value is implied.

IV. ELEMENT BEAM PATTERNS

For completeness, we present the derivation of the element radiation pattern in free space and the pattern of the element over an infinite planar ground screen using the method of images. Thus, a reference point for analyzing the effect of a finite ground screen on the beam pattern is established. The finite ground screen calculation is then presented in detail.

A. Beam Pattern of a Free-Space Element

The beam pattern calculation for a free-space element is straightforward, if some reasonable assumptions are made. The element is assumed to be constructed of infinitely-thin, perfectly conducting wire. The infinitely thin assumption is valid since the actual diameter (0.0254 m) is much less than a wavelength. Also, aluminum's very high conductivity at 216.980 MHz makes the perfect-conductor assumption reasonable. Thus, the element can be viewed as two ideal quarter-wave monopoles with appropriate relative current phases and relative positions.

The ideal quarter-wave monopole far-field beam pattern is calculated using well-known methods found in many introductory antenna theory books[1,2]. Consider a quarter-wave monopole oriented in the z direction with a current distribution ($I(z)$) given as

$$I(z) = I_o \sin\left(\frac{\pi}{2} - kz\right) \qquad 0 \le z \le \frac{\lambda}{4} , \tag{8}$$

where

$$k \equiv \frac{2\pi}{\lambda} , \tag{9}$$

and $z = 0$ is the feedpoint of the antenna. The electric field in spherical coordinates is then given by

$$\mathbf{E}(r, \theta, \phi) = E_o \frac{e^{-ikr}}{r} \frac{(e^{i\frac{\pi}{2}\cos\theta} - i\cos\theta)}{\sin\theta}\, \widehat{\mathbf{\Theta}}\,, \tag{10}$$

where

$$E_o \equiv I_o \frac{i\omega\mu}{4\pi k}, \tag{11}$$

$$\omega \equiv 2\pi f\,, \tag{12}$$

$$i \equiv \sqrt{-1}\,, \tag{13}$$

and μ is the magnetic permeability of the ambient medium. Implicit in this paper is an e^{iwt} time dependency associated with the currents and the electric fields.

If the mutual impedance between monopoles is neglected, the electric field for each monopole of the NAVSPASUR dipole element can be found by performing two different rotational coordinate transformations on Eq. (10). The superposition of these two expressions is then the desired far-field pattern. The z direction is chosen to be vertical, the x direction west, and the y direction south. Therefore, the x-z plane contains the dipole. The current phase difference between the monopoles is 180°. The electric field expression is then

$$\begin{aligned}
\mathbf{E}(r, \theta, \phi) = {} & E_o \frac{e^{-ikr}}{r} \frac{e^{-i\frac{\pi}{2}(\sin\delta\cos\theta + \cos\delta\sin\theta\cos\phi)} + i(\sin\delta\cos\theta + \cos\delta\sin\theta\cos\phi)}{1 - (\sin\delta\cos\theta + \cos\delta\sin\theta\cos\phi)^2} \\
& \times \left[(\sin\delta\sin\theta - \cos\delta\cos\theta\cos\phi)\,\widehat{\mathbf{\Theta}} + \cos\delta\sin\phi\,\widehat{\mathbf{\Phi}}\right] \\
& + E_o \frac{e^{-ikr}}{r} \frac{e^{-i\frac{\pi}{2}[\sin\delta\cos\theta - \cos\delta\sin\theta\cos\phi]} + i(\sin\delta\cos\theta - \cos\delta\sin\theta\cos\phi)}{1 - (\sin\delta\cos\theta - \cos\delta\sin\theta\cos\phi)^2} \\
& \times \left[-(\sin\delta\sin\theta + \cos\delta\cos\theta\cos\phi)\,\widehat{\mathbf{\Theta}} + \cos\delta\sin\phi\,\widehat{\mathbf{\Phi}}\right],
\end{aligned} \tag{14}$$

where δ is the droop angle of 55°, $\widehat{\mathbf{\Theta}}$ is unit vector in the theta direction, and $\widehat{\mathbf{\Phi}}$ is the unit vector in phi direction.

The resultant x-z (east-west-vertical) plane directivity pattern is sh wn in Fig. 2.

B. Infinite Ground Screen Pattern

A common method of simplifying radiation pattern calculations for elements with ground screens is the use of the infinite planar ground screen assumption. The infinite ground screen beam pattern calculation can be performed by the method of images. The method of images allows the infinite ground screen to be replaced by an image dipole. This dipole lies the same distance below the ground screen as the real dipole lies above, and has a current distribution 180° out of phase with that of the real dipole. The electric field above the ground plane is then a superposition of the real dipole field and the image dipole field. The calculation is then no more difficult than the free element pattern calculation, and only slightly more tedious. The infinite ground screen assumption works well for large ground screens extending many wavelengths away from an element. For smaller ground screens this is not the case, but unfortunately the difficulty of finite ground screen calculations often dictates that this assumption be used anyway.

Applying the method of images, the total far electric field expression becomes

$$
\begin{aligned}
\mathbf{E}(r,\theta,\phi) = {} & E_o \frac{e^{-ikr}}{r} \frac{e^{-i\frac{\pi}{2}(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)} + i(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)}{1-(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)^2} \\
& \times \left[(\sin\delta\sin\theta-\cos\delta\cos\theta\cos\phi)\,\widehat{\Theta}+\cos\delta\sin\phi\,\widehat{\Phi}\right] \\
& + E_o \frac{e^{-ikr}}{r} \frac{e^{-i\frac{\pi}{2}(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)} + i(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)}{1-(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)^2} \\
& \times \left[-(\sin\delta\sin\theta+\cos\delta\cos\theta\cos\phi)\,\widehat{\Theta}+\cos\delta\sin\phi\,\widehat{\Phi}\right] \\
& - E_o \frac{e^{-ikr}}{r} \frac{e^{i\frac{\pi}{2}(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)} - i(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)}{1-(\sin\delta\cos\theta-\cos\delta\sin\theta\cos\phi)^2} \\
& \times e^{-i(\frac{3\pi}{2})\cos\theta}\left[-(\sin\delta\sin\theta+\cos\delta\cos\theta\cos\phi)\,\widehat{\Theta}+\cos\delta\sin\phi\,\widehat{\Phi}\right] \\
& - E_o \frac{e^{-ikr}}{r} \frac{e^{i\frac{\pi}{2}(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)} - i(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)}{1-(\sin\delta\cos\theta+\cos\delta\sin\theta\cos\phi)^2} \\
& \times e^{-i(\frac{3\pi}{2})\cos\theta}\left[(\sin\delta\sin\theta-\cos\delta\cos\theta\cos\phi)\,\widehat{\Theta}+\cos\delta\sin\phi\,\widehat{\Phi}\right]. \qquad (15)
\end{aligned}
$$

Figure 3 shows the resultant x-z (east-west-vertical) plane directivity pattern.

C. Finite Ground Screen Pattern

The calculation of the radiation pattern of an element over a finite ground screen in the form of a wire grid is a formidable problem, for which numerical methods are needed. The numerical method chosen for this calculation is called the *method of moments*.

1. METHOD OF MOMENTS

a. Fundamental Concepts

The *method of moments*[1] is a mathematical procedure for obtaining matrix equations from a set of linear equations. It was first used for electric field problems in 1967 by R. F. Harrington[3], and has evolved into a very powerful analytical technique for solving a variety of problems in electromagnetic theory. The technique is particularly useful for problems dealing with electromagnetic scattering from wire grids.

The basic concept of this method can be seen with the treatment of electromagnetic radiation incident on a perfectly conducting wire of length L. The perfectly conducting wire requires both the electric field in the interior and the tangential component of the electric field at the surface of the wire be equal to zero. Therefore, on the surface the tangential component of the scattered electric field $\mathbf{E}_{||}^{s}$ must cancel the tangential component of the incident electric field $\mathbf{E}_{||}^{i}$, or

$$\mathbf{E}_{||} = \mathbf{E}_{||}^{s} + \mathbf{E}_{||}^{i} = 0 \,. \tag{16}$$

This requirement can be reiterated in terms of the induced current distribution $I(z')$ on the wire as

$$\frac{1}{i\omega\epsilon_0}\int_{-\frac{L}{2}}^{\frac{L}{2}} I(z')\left[\frac{\partial^2\psi(z,z')}{\partial z^2} + k^2\psi(z,z')\right] dz' + E_z^i(z) = 0 \,, \tag{17}$$

where $E_z^i(z)$ is the z component of the incident electric field, $\psi(z,z')$ is the free space Green function given by

$$\psi(z,z') \;=\; \frac{e^{ikR}}{4\pi R} \,, \tag{18}$$

and R is the distance between the observation point (x,y,z) and the source point (x',y',z'). For a thin wire of radius a $(a \ll \lambda, a \neq 0)$

$$R = \sqrt{(z - z')^2 + a^2}\,. \tag{19}$$

Equation (17) can be written in the form

$$\int_{-\frac{L}{2}}^{\frac{L}{2}} I(z')K(z, z')\, dz' = -E_z^i(z)\,. \tag{20}$$

The current distribution $I(z')$ can be approximated using a series of expansion functions F_n

$$I(z') \approx \sum_{n=1}^{N} I_n F_n(z')\,. \tag{21}$$

Substituting Eq. (21) into the Eq. (20) yields

$$\int_{-\frac{L}{2}}^{\frac{L}{2}} \sum_{n=1}^{N} I_n F_n(z')K(z, z')\, dz' \approx -E_z^i(z)\,. \tag{22}$$

For a simple case, F_n can be selected as a stairstep approximation to the actual current distribution, in which the wire is divided into N segments of equal length Δz and each segment supports a constant current. Then,

$$F_n(z') = \begin{cases} 1 & \text{for } z_n - \Delta z/2 \le z' \le z_n + \Delta z/2\,; \\ 0 & \text{otherwise.} \end{cases} \tag{23}$$

Here z_n is the coordinate of the midpoint of segment n. As N approaches infinity Δz approaches zero, and Eq. (21) approaches a true equality.

This choice of expansion functions allows the summation and I_n to be brought out of the integral in Eq. (22). The equality can then be enforced at the midpoint of each segment. Thus,

$$\sum_{n=1}^{N} I_n \int_{z_n - \frac{\Delta z}{2}}^{z_n + \frac{\Delta z}{2}} K(z_m, z')\, dz' \approx -E_z^i(z_m)\,, \qquad m = 1, 2, 3, ..., N \tag{24}$$

and the integral can be determined and defined as

$$f(z_m, z_n') \equiv \int_{z_n - \frac{\Delta z}{2}}^{z_n + \frac{\Delta z}{2}} K(z_m, z')\, dz' \,. \tag{25}$$

An analogy to network equations can be made with a set of generalized impedance, current, and voltage matrices. This leaves a set of N equations in the form

$$\sum_{n=1}^{N} Z_{mn} I_n = V_m \,, \tag{26}$$

where

$$Z_{mn} = f(z_m, z_n') \,, \tag{27}$$

and

$$V_m = -E_z^i(z_m) \,. \tag{28}$$

Expressed in matrix form, Eq. (26) becomes

$$\begin{pmatrix} Z_{11} & Z_{12} & \dots & Z_{1N} \\ Z_{21} & Z_{22} & \dots & Z_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ Z_{N1} & Z_{N2} & \dots & Z_{NN} \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ \vdots \\ I_N \end{pmatrix} = \begin{pmatrix} V_1 \\ V_2 \\ \vdots \\ V_N \end{pmatrix}, \tag{29}$$

from which I_n can be determined by matrix inversion.

Once I_n is known for each n, the far field can be obtained by summing the contributions from the individual segments and adding them to the far field from the original source.

This method, called point matching, can be used to determine the far field to a satisfactory degree of accuracy when N is large.

Point matching can be extended for use with an arbitrary thin-wire configuration. In particular, if l represents the coordinates of each position along a wire, then Eq. (20), in generalized form, becomes

$$\int_{l'} I(l') K(l, l')\, dl' = -E_l^i(l) \,. \tag{30}$$

Using the method of moments, with the current approximated by N expansion functions, Eq. (30) becomes

$$\sum_{n=1}^{N} I_n \int_{l'} F_n(l')K(l,l')\, dl' = -E_l^i(l)\,. \tag{31}$$

b. Piecewise Sinusoidal Galerkin Method

Although the point matching method is useful for some problems, its convergence is rather slow. Using different expansion functions can speed up the convergence. One common set of expansion functions is the piecewise sinusoid, which is given as

$$F_n(l) = \frac{\sin k(l - l_{n-1})}{\sin k(l_n - l_{n-1})} \qquad l_{n-1} \le l \le l_n\,, \tag{32a}$$

and

$$F_n(l) = \frac{\sin k(l_{n+1} - l)}{\sin k(l_{n+1} - l_n)} \qquad l_n \le l \le l_{n+1}\,, \tag{32b}$$

where l_n is the terminal of two connected segments.

A relatively small number of these overlapping sinusoids suffices to fit a typical current distribution quite well. However, in most cases some error will exist, because the electric field created by the current approximation does not cancel the incident electric field at all points along the wire. In other words

$$\sum_{n=1}^{N} I_n f(l,l'_n) + E_l^i(l) = R(l) \neq 0\,, \tag{33}$$

where $R(l)$ is the residual. By forcing the residual, averaged over the segment, to zero along each segment, the current approximation may be better. This may be further improved by weighting the errors over that segment. Again, N equations are formed as

$$\int_{\Delta l_m} W_m(l) \sum_{n=1}^{N} I_n f(l,l'_n)\, dl \;+\; \int_{\Delta l_m} W_m(l) E_l^i(l)\, dl = 0\,, \qquad m = 1,2,3,\ldots,N\,, \tag{34}$$

or

$$\sum_{n=1}^{N} I_n \int_{\Delta l_m} W_m(l) f(l, l'_n)\, dl \;+\; \int_{\Delta l_m} W_m(l) E_l^i(l)\, dl = 0\,, \qquad m = 1, 2, 3, \ldots, N\,, \tag{35}$$

where the W_m represent the weighting functions. Although the weighting functions can be chosen to be anything, they are often chosen to be the same as the expansion functions. This approach is known as the *Galerkin* method.

The weighting functions are also known as testing functions, for reasons related to the physical interpretation of Eqs. (34) and (35). This physical interpretation is related to the physical observable *reaction*. Reaction was first introduced in 1954 by V. H. Rumsey because the parameter that is measured at the terminals of an antenna is not the electric field, but the reaction between the receiver antenna and the transmitter antenna[5]. The reaction between antenna (or source) a and antenna (or source) b is given as

$$\langle a, b\rangle = \iiint_{V_b} [\mathbf{E}(a)\cdot\mathbf{J}(b) \;-\; \mathbf{H}(a)\cdot\mathbf{K}(b)]\, dv\,, \tag{36}$$

where $\mathbf{E}(a)$ and $\mathbf{H}(a)$ are the electric and magnetic fields radiated by source a, and $\mathbf{J}(b)$, $\mathbf{K}(b)$ are the electric and magnetic current distributions of source b, respectively. Also,

$$\langle a, b\rangle = \langle b, a\rangle\,. \tag{37}$$

For a perfectly conducting wire scatterer with an arbitrary configuration, the interior electric field vanishes. If the wires are replaced by the appropriate surface current densities, the internal and external fields do not change. Thus, an equivalent configuration exists. Test sources can be placed in the interior of these surface currents. The reaction of test source b with the scatterer a can be considered. If the test source and the scatterer are non-magnetic thin wires, then

$$\langle a, b\rangle = \int_l \mathbf{I}_b(l)\cdot\mathbf{E}_a(l)\, dl = 0\,. \tag{38}$$

The reaction is zero, since $\mathbf{E}_a$ is zero in the interior of the wire scatterer. Substituting for $\mathbf{E}_a$, as represented by Eq. (30), into Eq. (38) yields

$$\sum_{n=1}^{N} I_n \int_l dl\, I_b(l) \int_{l'} F_n(l') K(l, l') dl' + \int_l E_l^i(l)\, dl = 0\,. \tag{39}$$

Now, if the expansion functions are chosen to be piecewise sinusoids, a judicious choice of test source is a short dipole with unit input current. This choice (i.e. test dipoles) forces the test source to have a piecewise sinusoid current distribution. If N of these test dipoles are placed in overlapping positions along the wire configuration, so that the sinusoid distribution corresponds with the expansion function distribution, N equations are generated as

$$\sum_{n=1}^{N} I_n \int_{\Delta l_m} dl\, I_m(l) \int_{\Delta l'_n} F_n(l')K(l,l')\, dl' + \int_{\Delta l_m} E_l^i(l)\, dl = 0\,, \qquad m = 1,2,3,\ldots N\,, \tag{40}$$

where $I_m(l)$ is the function defined by Eq. (32).

Equation (40) takes the same form as Eq. (35). Therefore, the physical interpretation of Eq. (35) can be seen, if $W_m(l)$ represents $I_m(l)$ as the reaction between the test dipoles and the wire segments.

Equation (40) represents the particular method of moments known as the *piecewise sinusoidal Galerkin method*[1,4]. Each I_n is again found from the matrices represented by Eq. (29), where

$$Z_{mn} = \int_{\Delta l_m} dl\, I_m(l) \int_{\Delta l'_n} F_n(l')K(l,l')\, dl'\,, \tag{41}$$

and

$$V_m = -\int_l E_l^i(l)\, dl\,. \tag{42}$$

In practice, a particular wire configuration is mathematically divided into straight segments. Past experience has shown that the segment length should not be greater than $\lambda/4$. Test dipoles are then placed inside the wire over the entire configuration with each monopole corresponding to a segment. (The dipoles are not necessarily straight. The angle between the monopoles corresponds to the angle between connecting segments.) The current distribution must be represented so all possible current paths are allowed. Interior segments will contain overlapping monopoles, and segments at wire intersections may contain four monopoles, due to overlapping test dipoles. The current must fall to zero at points when a wire is terminated. Therefore, a segment at the termination of the wire contains only one monopole. In other words, each segment contains one monopole for each possible current path to or from that segment.

Once each I_n is known, the scattered far field can be determined by summing the contributions from each segment. For example, a typical segment of length d with endpoints (x_1, y_1, z_1) and (x_2, y_2, z_2), oriented by the angles α, β, and γ with respect to the three principal axes x, y, and z, respectively, has a current distribution represented by two overlapping sinusoids and expressed as

$$I(\ell) = \frac{I_1 \sin k(d-\ell) + I_2 \sin k\ell}{\sin kd}, \tag{43}$$

where $0 \le \ell \le d$ is the distance along the segment. The far field components are then expressed as

$$E_\theta = (\cos\alpha \, \cos\theta \, \cos\phi - \cos\beta \, \cos\theta \, \sin\phi - \cos\gamma \, \sin\theta) \, E_\ell \, , \tag{44}$$

$$E_\phi = (-\cos\alpha \, \sin\phi + \cos\beta \cos\phi) E_\ell \, , \tag{45}$$

where

$$E_\ell = \frac{\eta e^{-ikr}}{4\pi r(1-g^2)\sin kd} \Big[(e^{ikgd} - g \sin gd - \cos gd) I_1 e^{ikf_1} + (e^{-ikgd} + g \sin kd - \cos kd) I_2 e^{ikf_2} \Big] \, , \tag{46}$$

$$f_1 = x_1 \sin\theta \, \cos\phi + y_1 \sin\theta \, \sin\phi + z_1 \cos\theta \, , \tag{47}$$

$$f_2 = x_2 \sin\theta \, \cos\phi + y_2 \sin\theta \, \sin\phi + z_2 \cos\theta \, , \tag{48}$$

and

$$g = \cos\alpha \, \sin\theta \, \cos\phi + \cos\beta \, \sin\theta \, \sin\phi + \cos\gamma \, \cos\theta \, . \tag{49}$$

The scattered far field from each segment is added to the original incident far field of the radiating element to determine the total far field.

2. APPLICATION TO NAVSPASUR

Since the ground screen below the NAVSPASUR element is a wire-grid ground screen, the method of moments is the logical approach to calculating the radiation pattern. In order to solve Eq. (29) for the current distribution, a number of unknowns need to be determined.

First, the north-south length of ground screen to use in the calculation must be determined. Computer memory limitations do not allow the calculation of the scattering from the full ground screen. Fortunately, the entire length of the ground screen is not needed for an accurate calculation of the element radiation pattern. Very little contribution to the beam pattern comes from sections of the screen that are sufficiently far away from the element. A sequence of calculations involving a systematic north-south lengthening of the ground screen has been performed to determine the distance at which the ground screen can be safely truncated. This assures that the north-south length used in the final calculation is a good approximation to the very long ground screen.

The incident field which originates from the transmitter element must also be determined. Although the far field approximation is given in Section II, the near electric field is clearly required in this case. Although the assumptions used in Section II (excluding the far-field assumption) are still valid, the calculation is more difficult. An expression for the near field of an arbitrary monopole has been determined in a manner similar to one used in the literature[6]. This expression, in polar coordinates, is

$$\mathbf{E}(r,\theta,\phi) = E_\circ \left[\frac{e^{-ikr_1}}{r_1}\left(\frac{-L}{r}\right) + \frac{e^{-ikr}}{r}\left(\frac{\sin kL}{kr}\right)\right]\hat{\mathbf{r}}$$
$$+ \frac{E_\circ}{\sin\theta}\left[\frac{e^{-ikr_1}}{r_1}\left(1 - \frac{L\cos\theta}{r}\right) - \frac{e^{ikr}}{r}(\cos kL + i\cos\theta\,\sin kL)\right]\hat{\Theta}, \quad (50)$$

where L is the length of the monopole, r is the distance from the feedpoint of the monopole, r_1 is the distance from the opposite end of the monopole, and $E_\circ$ is defined by Eq. (11).

For a quarter-wave monopole, Eq. (50) reduces to

$$\mathbf{E}(r,\theta,\phi) = \frac{E_\circ}{kr}\left(\frac{e^{-ikr}}{r} - \frac{\pi}{2}\frac{e^{-ikr_1}}{r_1}\right)\hat{\mathbf{r}}$$
$$+ \frac{E_\circ}{\sin\theta}\left[\frac{e^{-ikr_1}}{r_1}\left(1 - \frac{\pi}{2kr\cos\theta}\right) - \frac{e^{ikr}}{r}\,i\cos\theta\right]\hat{\Theta}. \quad (51)$$

The final expression used to obtain the NAVSPASUR element's free-space near field has been derived from the quarter-wave monopole expression via coordinate transformation and superposition, and is given by

$$
\begin{aligned}
\mathbf{E}(r,\theta,\phi) = E_\circ & \left\{ \frac{\pi}{2kr} \left[\frac{e^{-ikr_2}}{r_2} - \frac{e^{-ikr_1}}{r_1} \right] \right\} \hat{\mathbf{r}} \\
& - E_\circ \left\{ \left[\frac{e^{-ikr}}{r} 2i \left(\sin^2\delta - \cos^2\delta \cos^2\phi \right) \right] \right. \\
& \quad + \left[\frac{e^{-ikr_1}}{r_1} - \frac{e^{-ikr_2}}{r_2} \right] \left[\sin\delta \cos\theta + \frac{\pi}{2kr} \sin\theta \cos\theta (\sin^2\delta - \cos^2\delta \cos^2\phi) \right] \\
& \quad \left. - \left[\frac{e^{-ikr_1}}{r_1} + \frac{e^{-ikr_2}}{r_2} \right] \left[\cos\delta \cos\theta \cos\phi + \frac{\pi}{2kr} \sin\delta \cos\delta \cos\phi (\sin^2\theta - cos^2\theta) \right] \right\} \\
& \quad \times \left[1 - (\sin\delta \cos\theta - \cos\delta \sin\theta \cos\phi)^2 \right]^{-1} \hat{\boldsymbol{\Theta}} \\
& - E_\circ \left\{ \frac{e^{ikr}}{r} 2i \cos\delta \sin\theta \cos\phi \right. \\
& \quad \left. - \left[\frac{e^{-ikr_1}}{r_1} + \frac{e^{-ikr_2}}{r_2} \right] \left[1 + \frac{\pi}{2kr} \sin\delta \cos\theta \right] \right\} \cos\delta \sin\phi \\
& \quad \times \left[1 - (\sin\delta \cos\theta - \cos\delta \sin\theta \cos\phi)^2 \right]^{-1} \hat{\boldsymbol{\Phi}} \,, \qquad (52)
\end{aligned}
$$

where r_1 and r_2 are the distances from the two outside ends of the dipole, given by

$$
r_1 = \sqrt{r^2 - \frac{r\lambda}{2} (\cos\delta \sin\theta \cos\phi + \sin\delta \cos\theta) + \frac{\lambda^2}{16}} \qquad (53)
$$

$$
r_2 = \sqrt{r^2 + \frac{r\lambda}{2} (\cos\delta \sin\theta \cos\phi + \sin\delta \cos\theta) + \frac{\lambda^2}{16}} \,. \qquad (54)
$$

Finally, the segment size and location must be determined. The ground screen must be divided into segments. The maximum segment length is dictated by the ground screen grid size and must also satisfy the $\lambda/4$ maximum length requirement. Test sources must also be chosen for the piecewise sinusoidal Galerkin method in accordance with standards set in the previous section.

As shown in Section III-B, an $N \times N$ matrix is created, in which N is the number of actual sources, or test dipoles. N has a practical upper limit, which is dependent on the available computer memory. N is proportional to both the segment length and ground screen size. First, an acceptable segment length must be determined. This will dictate the maximum ground screen length (the width is fixed at 3.66 m) which can be used in the calculation.

The segment length is determined empirically by starting with a particular value and monitoring the change in the pattern as the segment length is varied. As the segment length is decreased, the pattern should asymptotically approach the correct result. The maximum segment length that gives a pattern which varies insignificantly from the asymptotic pattern should be used. Once the segment length is determined, the overall ground screen length can be determined. Again, as the screen is lengthened, an asymptotic pattern is reached. With luck, this is reached before the computer memory is exhausted.

3. RESULTS

The east-west single element pattern shown in Fig. 4 has been obtained using the piecewise sinusoidal Galerkin method of moments. Calculations have shown that segments corresponding directly with the grid spacing are adequate to produce an accurate pattern. That is, a combination of segments of length 0.305 m (the east-west grid spacing) and 0.076 m (the north-south grid spacing) can be used.

For the current work, computer memory limitations prevented using a sufficiently long ground screen to reach the asymptotic north-south element pattern. Fortunately, the portion of the pattern which was still changing significantly as the ground screen was lengthened was the section near the horizontal in the north-south plane. Since the purpose of computing the element pattern is to allow for the calculation of the transmitter antenna pattern, this is not a serious problem. The electric field of the element at north-south angles near the horizontal does not contribute to the main lobe or near sidelobes of the transmitter antenna beam pattern, which are near vertical. The near sidelobes are the only sidelobes of significant strength.

The wire composition is unknown, but this is not a serious problem. Metals in general have a high conductivity at 216.980 MHz. So the perfect conductor approximation should be good.

The replacement ground screen will not affect the pattern significantly, since the width of the ground screen is the major factor in determining the east-west pattern. The bars were chosen for sturdiness. The narrow spacing was chosen to lessen the leakage of radiation through the ground screen. The present leakage is not a significant part of the total power, but is above safety limits.

4. DISCUSSION

Since the purpose of this calculation is to obtain a good model for the present NAVSPASUR elements, the calculation must be compared to the actual pattern. Although no information is available concerning a single element pattern, some direct information can be obtained from the transmitter antenna patterns. This is possible because the shape of the east-west element pattern is the same as that of the east-west antenna pattern. The north-south pattern, as well as any pattern in an off east-west plane, is altered when the element is incorporated into the antenna array, so no comparison between the calculated single element pattern with actual transmitter antenna patterns can be made. Thus these patterns have been ignored.

The 3 dB and 6 dB east-west beam widths of the single element calculation and the transmitter antennas can be compared. The reported 3 dB and 6 dB east-west antenna beam widths are 115° and 140°, respectively[7]. The calculated single element widths of 112° and 136° agree well with these values, and are certainly an improvement over the predictions for infinite ground screen widths of 112° and 170°.

As can be seen by comparing Figs. 3 and 4, the finite wire-grid ground screen pattern differs significantly from the infinite ground screen patterns at angles below 65° with respect to vertical. This is supported by the 6 dB beam width comparison. Therefore, the use of the infinite ground screen approximation in transmitter antenna model exaggerates system capabilities at low angles.

The finite ground screen calculation has been performed with the benefit of several assumptions. Many of these assumptions need not be made, but not making them increases the complexity of the calculation. Possible refinements to the calculation include accounting for the finite conductivity of the elements and the ground screen, the mutual inductance between the monopoles of the element, and the mutual inductance between the element and the ground screen.

The calculated pattern parameters agree well with the reported parameter values, so further refinements are expected to make only minor changes to the pattern. Also, these changes may not make the calculated pattern agree more closely with the actual pattern, since other factors cannot be accurately modeled. Among these are unknown deviations from reported structural specifications and the effect of the uneven terrain in the immediate area of the antennas. Therefore, the calculation has been terminated at this point.

The analytic tools developed for this calculation, can easily be used to perform calculations of other antenna elements positioned over finite wire-grid ground screens. In particular, the NAVSPASUR receiver element patterns can be calculated in the same manner, with only minor changes in our computer code. The code is discussed in more detail in the Appendix.

V. ANTENNA PATTERNS

The specifications quoted in Section II were used to calculate the transmitter antenna patterns. The element beam patterns are assumed to be identical for each element. Although the ground screen ends will have an effect on the beam patterns of elements near the ends of the antenna bays, given the small number of dipoles to which this condition applies, the overall effect on the antenna beam pattern is insignificant.

The far-field calculations of the antenna beam pattern are easily performed by convolving the element pattern with the array pattern. Here, the array pattern refers to the pattern resulting from an array of isotropic point sources fed in phase. However, near-field patterns must be produced via an element-by-element summation to obtain the electric field at any point in space. Use of the current distributions of the segments to calculate the patterns directly would consume unacceptable amounts of CPU time (about 60 CPU-days for a single pattern on a VAX 11/750!). Fortunately, the electric field expression of the element over the region of concern can be well approximated with two polynomial equations. This reduces the computation time considerably, making near-field calculations feasible.

A. Kickapoo Complex

The north-south near-field beam patterns of the Kickapoo complex are the patterns of greatest importance. Since the near field extends to about 15,000 km, most of the satellites detected by NAVSPASUR pass through the fence within this range. Several north-south power density patterns along with the electric field phase plots are shown for various satellite ranges in Figs. 5–13. The near-field power density patterns are plotted as power density versus angle with zero degrees representing vertical. The power density is plotted in units of dBm, or deciBels above or below 1 milliWatt. Note that the scale of the angle axis varies with range, to accommodate the varying width of the main lobe of the patterns.

The far-field (infinite range) pattern (Fig. 5) has the expected linear array shape, a main vertical lobe with sidelobes of decreasing power away from vertical. The range of 100,000 km is well within the far-field region and that pattern (Fig. 6a) closely resembles the far-field pattern.

Only far-field parameters of the transmitter antennas are reported in NAVSPASUR handbooks[7,8]. Our calculated north-south 3 dB far-field beamwidth of 0.021° agrees with the reported value, as does the first sidelobe level of −13 dB. The calculated directivity of 40.1 dB is slightly less than the reported gain value of 42 dB. The east-west beamwidths are the same as the element pattern, as discussed in Section IV-C-4.

Although the near-field parameters are not reported, the calculation is straight-forward and we are confident of our results. As the transmitter antenna range falls within the near-field region the sidelobe gaps start to fill in and the sidelobe levels increase relative to the main lobe (Figs. 7a–9a). Well inside the near-field region (see e.g. 1,000 km - Fig. 10a),

the peak power density is no longer in the vertical direction and the main lobe merges with the sidelobes to create a wider main lobe whose characteristics are not well-defined. The width and shape of this main lobe is range dependent and the power density can have variations greater than 3 dB (Figs. 10a–13a). The 3 dB beamwidth ranges from its far-field width of 0.021° to about 1.8° at a range of 100 km where it is no longer well-defined.

The asymmetry in the near-field patterns is produced by the road gap in bay #8. This asymmetry disappears completely when bay #8 is treated as a normal bay, as can be seen by comparing Fig. 10 with Fig 15.

The power density in the near field has a different range dependency than in the far field. This is shown in Fig. 14, which compares power density in the vertical direction of the Kickapoo complex with a theoretical point source antenna of the same power. This plot show three distinct regions, corresponding roughly with the three decades. The far-field region corresponds roughly to the third decade (10,000–100,000 km). In this region the range dependency of the antennas is the same. The middle decade (1,000–10,000 km) is a transition region, where both near-field and far-field characteristics are seen in the power density patterns of the Kickapoo complex and deviations from the point source occur. The vertical power density at 1,000 km is actually less than that at 2,000 km. Below 2,000 km the vertical direction no longer represents the peak power density. The first decade shows a great difference between the two power densities, although both show a well-behaved range dependency.

The electric field phase patterns also exhibit a difference between the far-field and near-field patterns. For ranges greater than 5,000 km the phase is relatively constant across the main lobe. Inside this range the phase reaches a minimum at vertical but exhibits a sharp rise on both sides of the vertical.

Clearly, the near-field patterns reveal that any model of the Kickapoo complex must include near-field effects. The assumption of far-field behavior may easily lead to false conclusions. Specifically, below 1,000 km power densities are overestimated and beamwidths are underestimated by over an order of magnitude. Also below 5,000 km phase variations across the beam are important. Such phase variations are not expected to affect the Doppler determinations for Kickapoo-illuminated satellite returns (since they are symmetric), but are believed to be the source of the documented chirp bias associated with Kickapoo observations[9].

B. Gila River

The Gila River antenna near field extends only to about 360 km. Only a few satellites are observed inside this range. Figure 16 shows the 100 km beam pattern, which can be compared to the far-field pattern, shown in Fig. 17.

The far-field 3 dB beamwidth and the first sidelobe levels agree well with the reported values of 0.14° and −13 dB, respectively. As with the Kickapoo complex, the calculated directivity of 31.9 is lower than the reported gain value of 34 dB.

C. Jordan Lake

The Jordan Lake antenna is slightly shorter than the Gila River antenna. Its near field extends to about 140 km, so its near-field plays the least significant role.

Jordan Lake's far-field pattern, shown in Fig. 18, has far-field 3 dB beamwidth and first side lobe levels which agree with the reported values 0.22° and −13 dB, respectively. Again, the calculated directivity value of 30.1 is less than the reported gain value of 32 dB.

D. Discussion

Our transmitter calculations clearly show that although the far-field approximation can be used for the Gila River and Jordan Lake transmitting antennas, near-field effects must be included in any model of the Kickapoo complex. We have developed such a model, which can be used as is, or incorporated into any NAVSPASUR system model.

The computer code, which is discussed in more detail in the Appendix, has been designed to provide versatility in antenna calculations. Among the variables which can be changed are range, angles, and complex element current.

Besides the antenna calculations detailed in the previous sections, antenna calculations which include random and systematic current phase and amplitude errors of individual elements or bays have been performed. We will report these results in a separate paper.

VI. CONCLUSIONS

We have performed an accurate, detailed calculation of the NAVSPASUR transmitter antenna element pattern, and have used this element pattern to obtain a set of near- and far-field beam patterns for the three NAVSPASUR transmitter antennas. These patterns represent the first detailed calculations of the performance of the NAVSPASUR transmitter antennas which properly account for the effects of the finite wire-grid ground screen.

The east-west patterns of the antennas are a significant improvement over patterns obtained with the assumption of an infinite ground plane. For angles near the horizon the infinite ground-plane assumption substantially overestimates the radiation intensity transmitted by the antennas.

The north-south beam patterns we have derived demonstrate that the Gila River and Jordan Lake transmitters can be accurately modeled using a far-field approximation. However, the far-field approximation is clearly inappropiate for the Kickapoo complex. At ranges below about 2,000 km the radiation intensity is significantly overestimated and

the beamwidth significantly underestimated using that assumption. Further, for ranges under 5,000 km near-field effects introduce significant phase variations across the beam. These phase variations may introduce systematic errors in the derived Doppler and chirp for targets illuminated by the Kickapoo transmitter.

We believe our NAVSPASUR transmitter antenna model provides a solid foundation for a broader NAVSPASUR system model. Our results can be incorporated into a NAVSPASUR system simulation program to provide a better understanding of the capabilities of the present system. Thus, informed decisions involving future enhancements to the NAVSPASUR system can be made.

Of further importance are the analytic tools which we have developed. An equally accurate and detailed model of the present receiver antennas can be obtained using these same tools. Also, the versatility of our computer code allows proposed changes to either the transmitter or receiver antennas to be easily and accurately evaluated.

APPENDIX–COMPUTER CODE

A. THINWIRE

THINWIRE.FOR is the FORTRAN code used to generate the current values I_n using the piecewise sinusoidal Galerkin method. Some of the subroutines in THINWIRE are adapted from a general code developed by Richmond in 1970[10]. THINWIRE incorporates Richmond's pertinent subroutines with subroutines we have developed, which are specific to the NAVSPASUR geometry.

Although THINWIRE was developed specifically for the element pattern, its flexible modular structure makes it convenient to substitute other subroutines for the present ones to generate the ground screen currents for other elements. The next logical use of this algorithm would be to calculate the NAVSPASUR receiver element patterns.

B. GEN_ARRAY

GEN_ARRAY is a versatile VAX/VMS command file which is short for general array program. This can generate a number of unknowns for a general array.

GEN_ARRAY has four variable parameters: P1, P2, P3, and P4. P1 is the name of the main FORTRAN program to be executed. P2 is the name of the element position file which holds the subroutine ELPOS. P3 is the name of the file containing the subroutine CURRNT. P4 is the file containing ELFFLD. GEN_ARRAY compiles, links, and runs the code, and assigns the appropriate FORTRAN unit numbers to the input and output files.

The main program determines the information that is extracted from the electric fields of the array of interest. It calls the subroutine ARRAY_SBR which inputs the range and two direction angles to point of interest and returns three complex components of the array's electric field. The main program then uses these values to determine the parameter of interest. A general purpose program named NEARFLD has been developed for the calculations presented in this paper.

NEARFLD computes power density at a selected range for the near-field beam patterns. The program is designed so that the otherwise useless range value of zero triggers a far-field calculation. NEARFLD also computes the electric field phase at the selected points in space.

ELPOS, CURRNT, and ELFFLD are the names of the subroutines called by the subroutine ARRAY_SBR. Each of these names represents a set of specific subroutines of the same subroutine name, but with unique file names. The limitations of FORTRAN prevent a variable name in the CALL statement. Therefore, GEN_ARRAY is used to maintain versatility by choosing the specific files of interest and linking them with the main program.

ELPOS provides the number of elements in the array and their positions. Two ELPOS files have been created with names ELPOS_TRNS and ELPOS_RCVR, which provide the information for the transmitters and receivers, respectively. Input files that are read by these subroutines allow one to choose the particular transmitter or receiver configuration. ELPOS_TRNS also provides for the special case of a single element array for element patterns.

CURRNT provides the amplitude and phase of the input current for each element in the array. Four subrountines have been created. CURRNT_STND provides a random distribution of complex current values to the array elements with a specified mean amplitude and phase. CURRNT_RBAY provides a random distribution of mean complex current values for the bays of the Kickapoo complex. However, all elements within a bay are excited in phase. CURRNT_SBAY provides systematic bay-to-bay amplitude and phase errors with all elements within a bay excited in phase. CURRNT_RSBY provides systematic bay-to-bay amplitude and phase errors, together with randomly distributed amplitudes and phases among the elements within the bay.

ELFFLD returns the far electric field values at a specified point in space for a particular element. Five subroutines have been created. ELFFLD_TRNS provides the far-field values for a transmitter element above a finite ground screen using the current values calculated by THINWIRE. ELFFLD_APRX provides the the transmitter element's far-field values using a polynomial approximation. The transmitter element pattern in free space and over and infinite ground plane are obtained by ELFFLD_AHHW and ELFFLD_AHGP, respectively. ELFFLD_RCVR provides the NAVSPASUR receiver element's far-field values for a receiver element above an infinite ground plane.

These subroutines allow one to the calculate patterns for a number of different situations. Certainly others can be created. The combination of the subroutines with THINWIRE's flexibility allows for quick and easy adaptation to a variety of specific configurations.

REFERENCES

1. W. L. Stutzman and G. A. Thiele, *Antenna Theory and Design*, Wiley, New York, 1981.

2. E. A. Wolff, *Antenna Analysis*, Wiley, New York, 1966.

3. R. F. Harrington, "Matrix Methods for Field Problems" *Proc. of the IEEE*, vol. 55, February 1967.

4. J. H. Richmond,"Radiation and Scattering by Thin-Wire Structures in the Complex Frequency Domain" National Technical Information Service, Springfield, VA 22151, NASA Contractor Report CR-2396, May 1974.

5. V. H. Rumsey,"The Reaction Concept in Electromagnetic Theory,"*Physical Review,* Ser. 2, vol. 94, June 1954.

6. R. Bechmann,"Calculation of Electric and Magnetic Field Strengths of Any Oscillating Straight Conductors," *Proc. of the IRE*, vol. 19, March 1931.

7. Naval Research Laboratory, Washington, DC, "Configuration Control Manual for the Navy Space Surveillance System," Contract: N178-8624, 15 July 1967.

8. Naval Space Surveillance System, "Handbook for Naval Space Surveillance System, Volume I: NAVSPASUR System Orientation," Naval Electronic Systems Command, 4600 Marriott Dr., North Charleston, SC, 1 July 1976.

9. S. H. Knowles, W. B. Waltman, and R. H. Smith, "Experimental Observations of Naval Space Surveillance Satellite Signals with an Out-of-Plane Receiving Station", NRL Memorandum Report 4831, 15 June, 1982.

10. J. H. Richmond, "Computer Program for Thin-Wire Structures in a Homogeneous Conducting Medium," National Technical Information Service, Springfield, VA 22151, NASA Contractor Report CR-2399, July 1973.

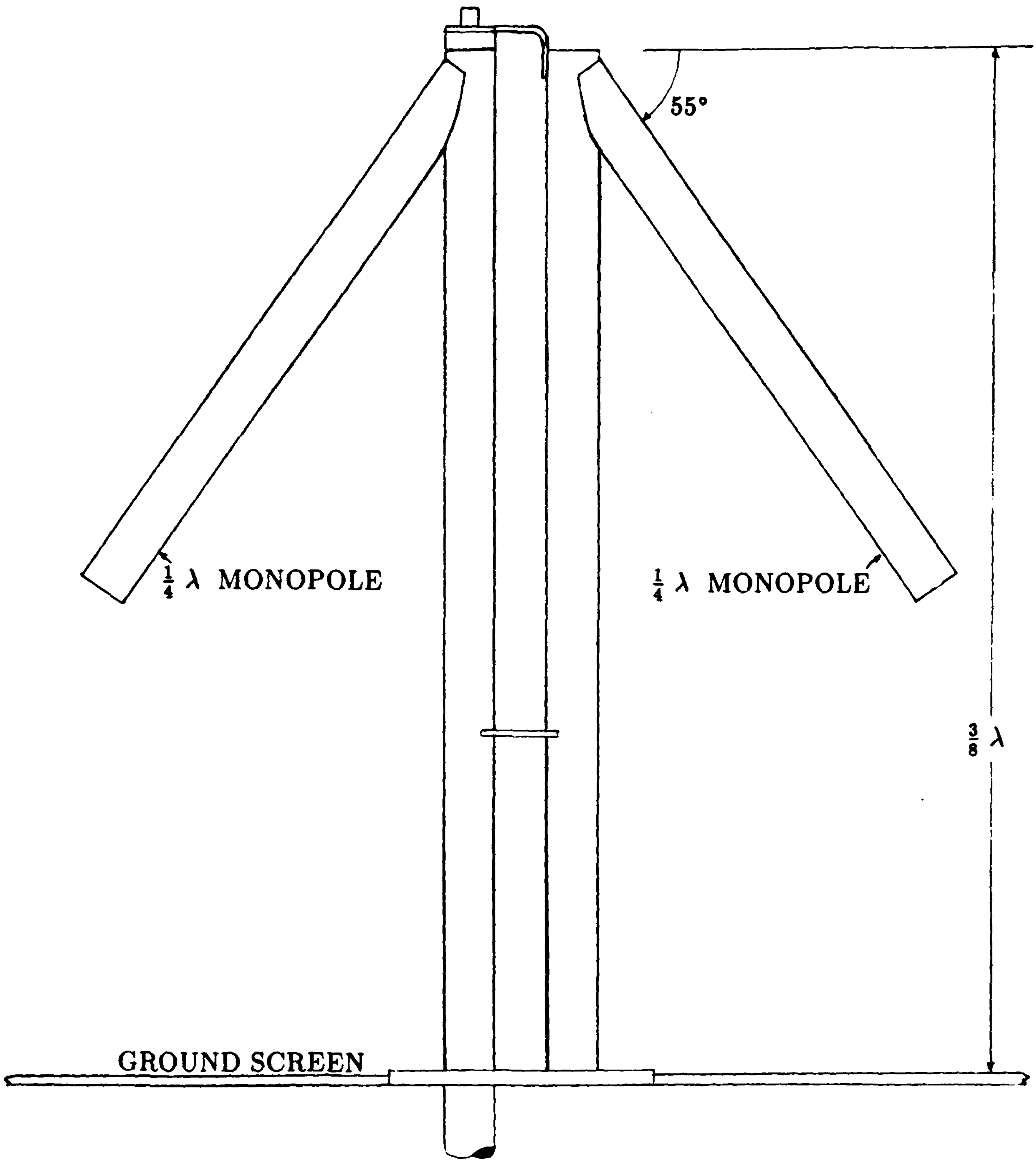

55°
$\frac{1}{4}\lambda$ MONOPOLE
$\frac{1}{4}\lambda$ MONOPOLE
$\frac{3}{8}\lambda$
GROUND SCREEN

Figure 1

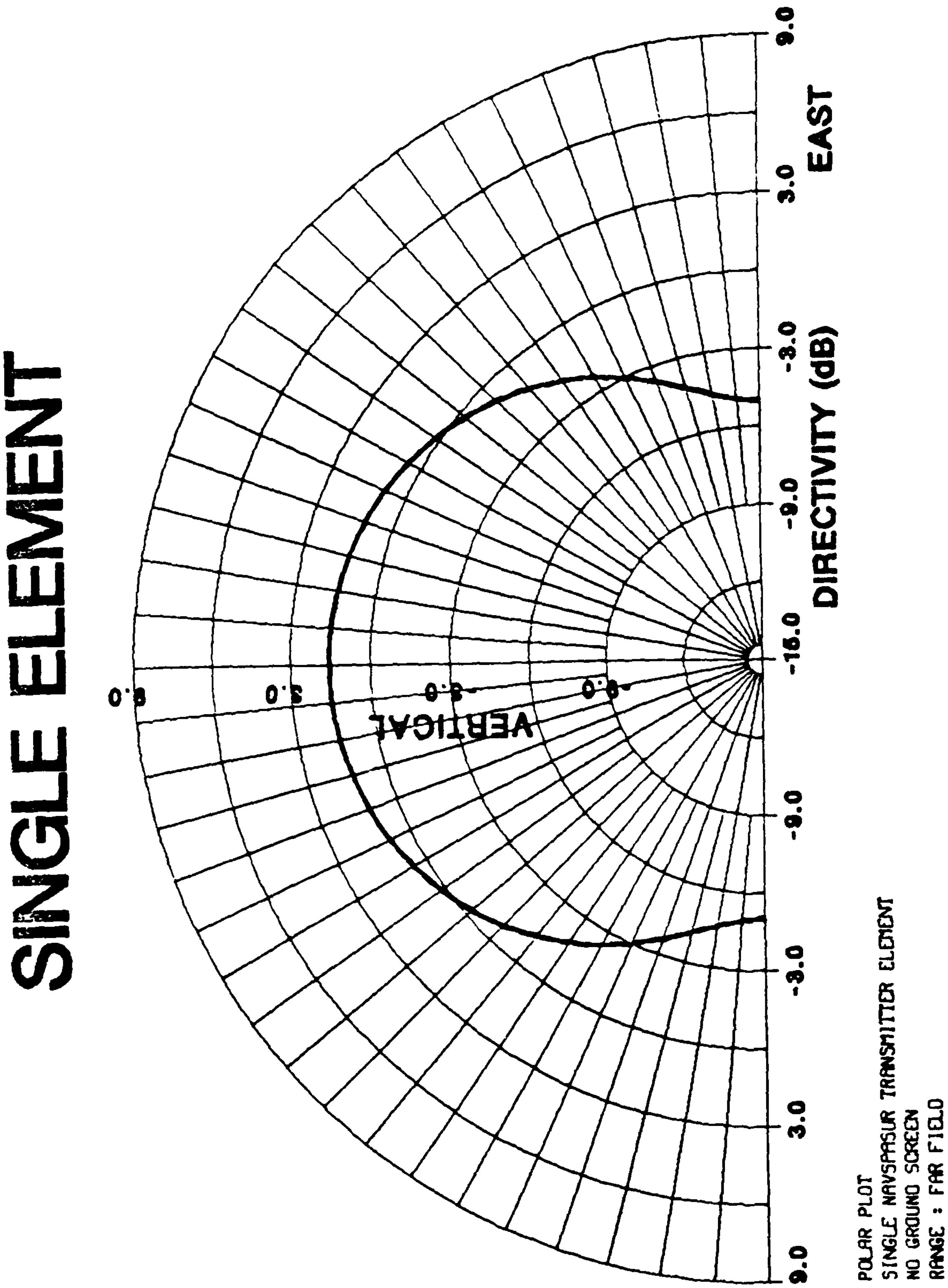

SINGLE ELEMENT
EAST
DIRECTIVITY (dB)
9.0
3.0
-3.0
-9.0
-15.0
VERTICAL
POLAR PLOT
SINGLE NAVSPASUR TRANSMITTER ELEMENT
NO GROUND SCREEN
RANGE : FAR FIELD

Figure 2

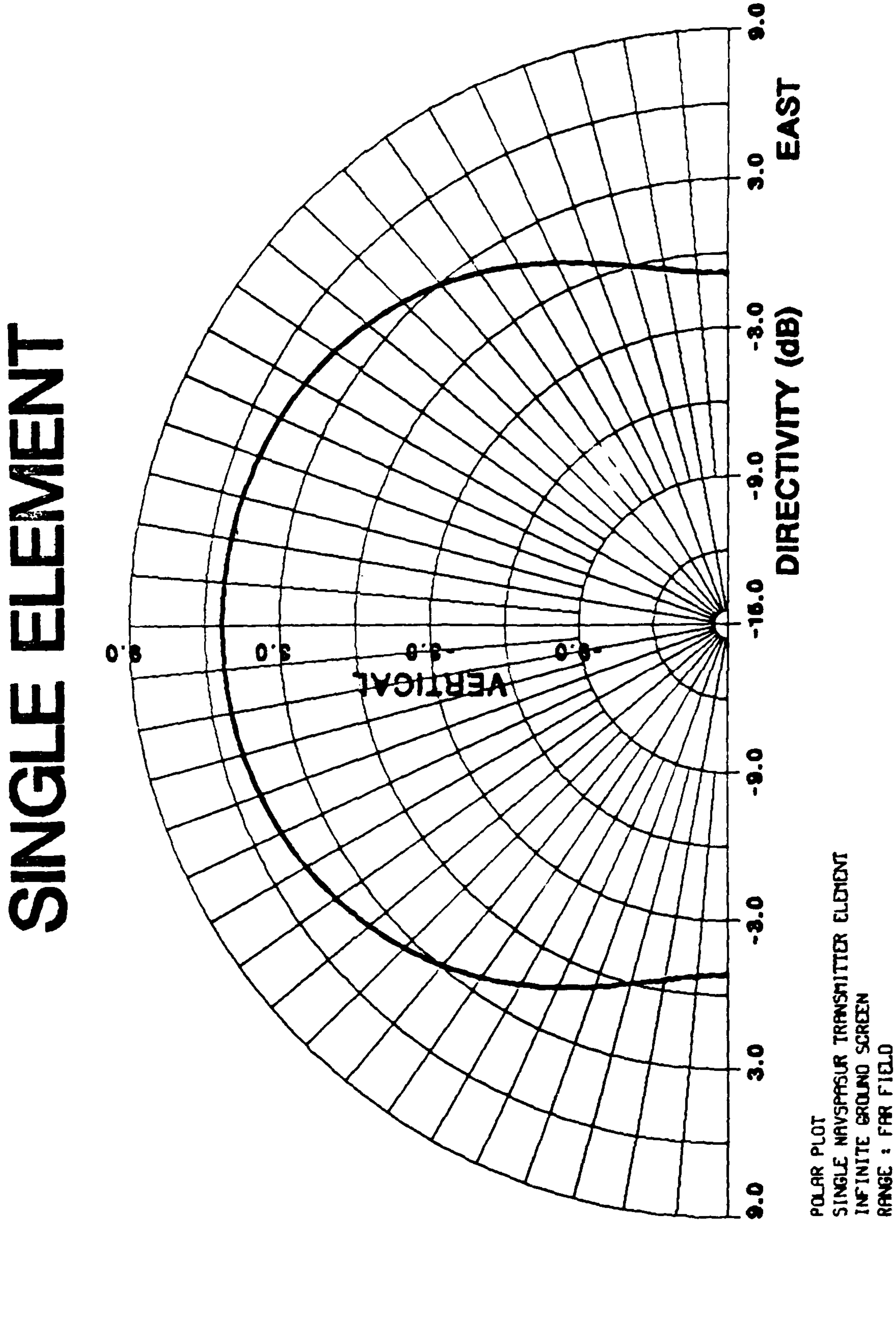

SINGLE ELEMENT
9.0
3.0
-3.0
-9.0
-15.0
-9.0
-3.0
3.0
9.0
EAST
DIRECTIVITY (dB)
VERTICAL
POLAR PLOT
SINGLE NAVSPASUR TRANSMITTER ELEMENT
INFINITE GROUND SCREEN
RANGE : FAR FIELD

Figure 3

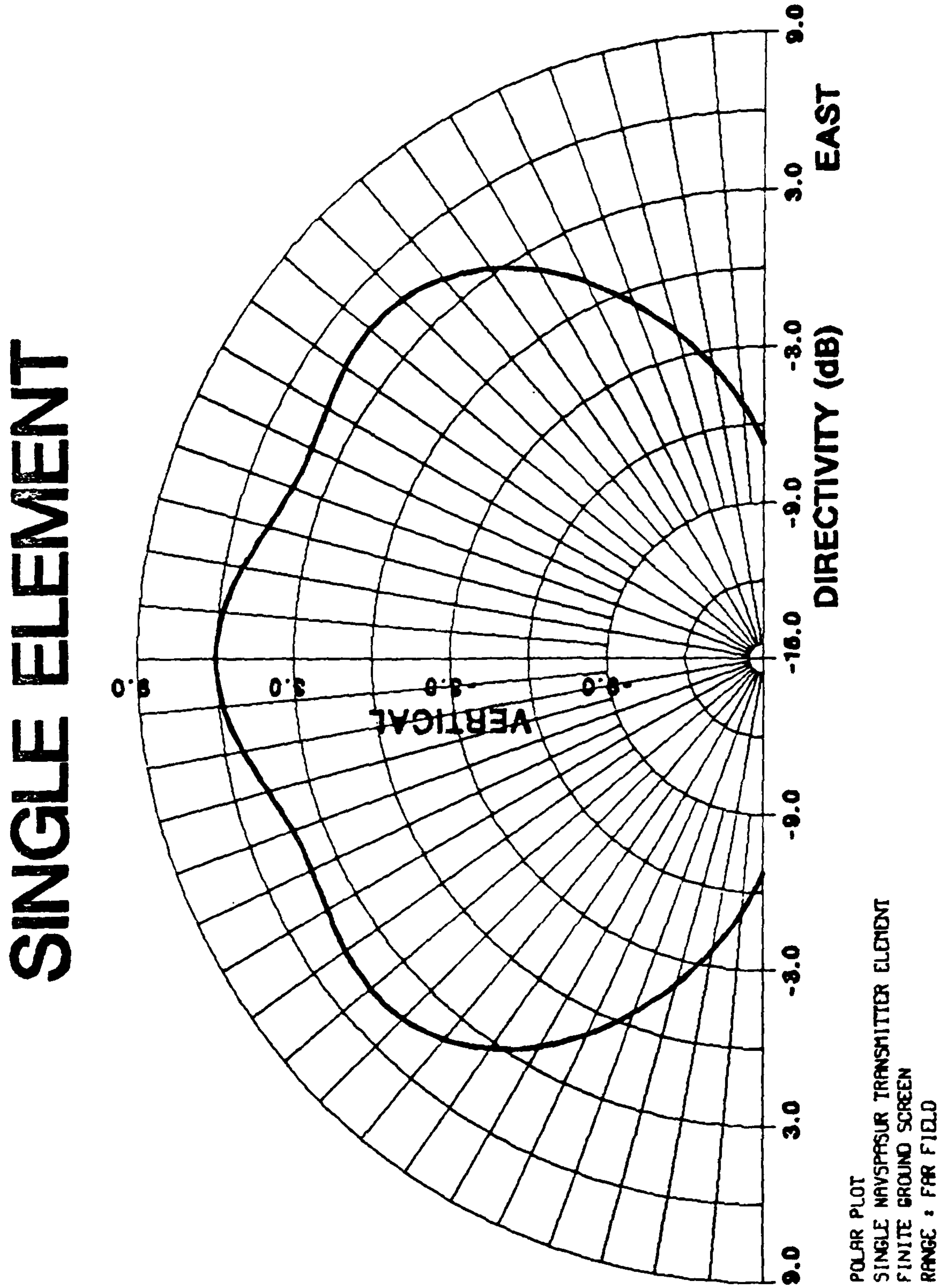

SINGLE ELEMENT
VERTICAL
EAST
DIRECTIVITY (dB)
POLAR PLOT
SINGLE NAVSPASUR TRANSMITTER ELEMENT
FINITE GROUND SCREEN
RANGE : FAR FIELD

Figure 4

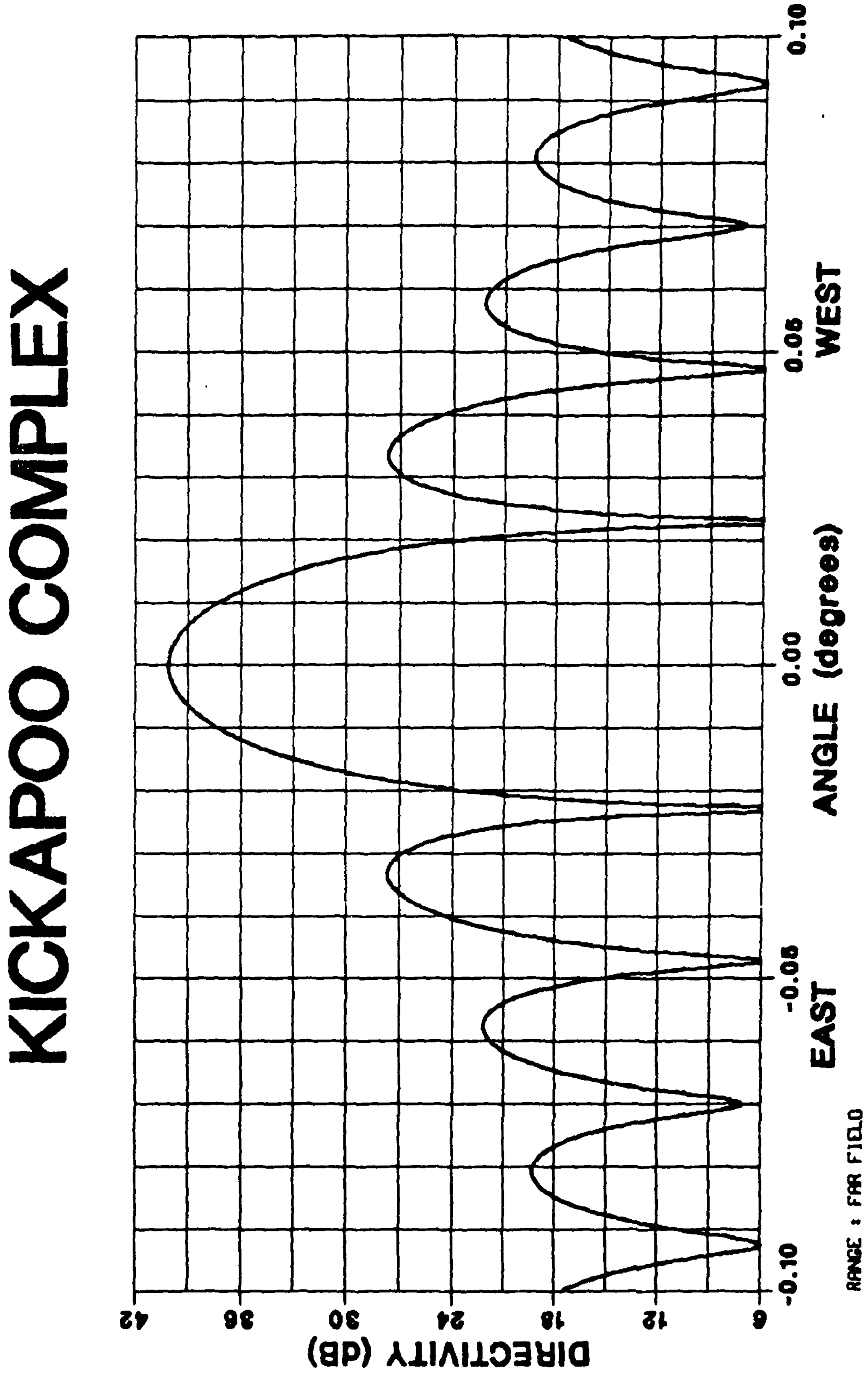

KICKAPOO COMPLEX
DIRECTIVITY (dB)
42
36
30
24
18
12
6
-0.10
-0.05
0.00
0.05
0.10
EAST
ANGLE (degrees)
WEST
RANGE : FAR FIELD

Figure 5

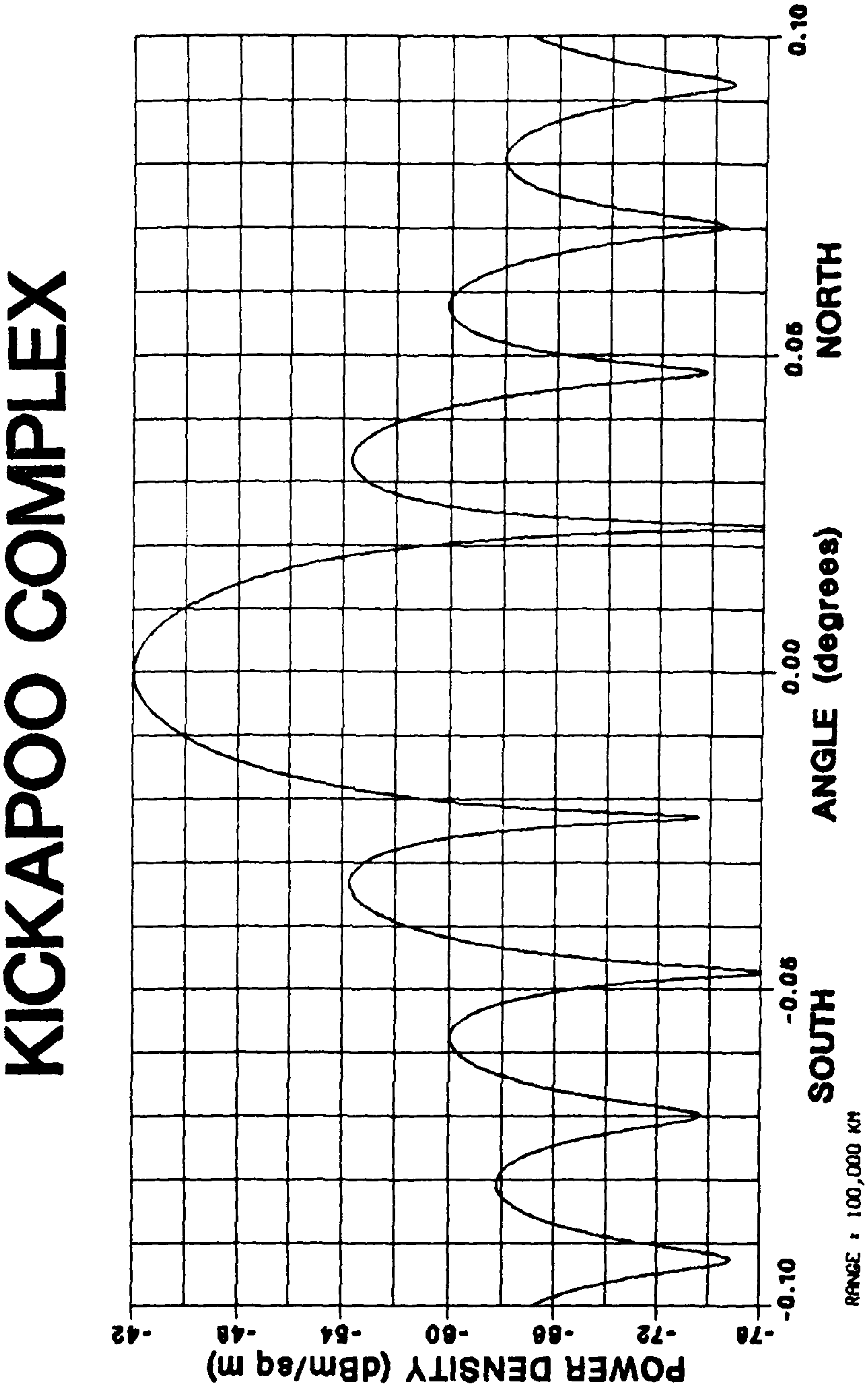

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-42
-48
-54
-60
-66
-72
-78
-0.10
-0.05
0.00
0.05
0.10
SOUTH
ANGLE (degrees)
NORTH
RANGE : 100,000 KM

Figure 6a

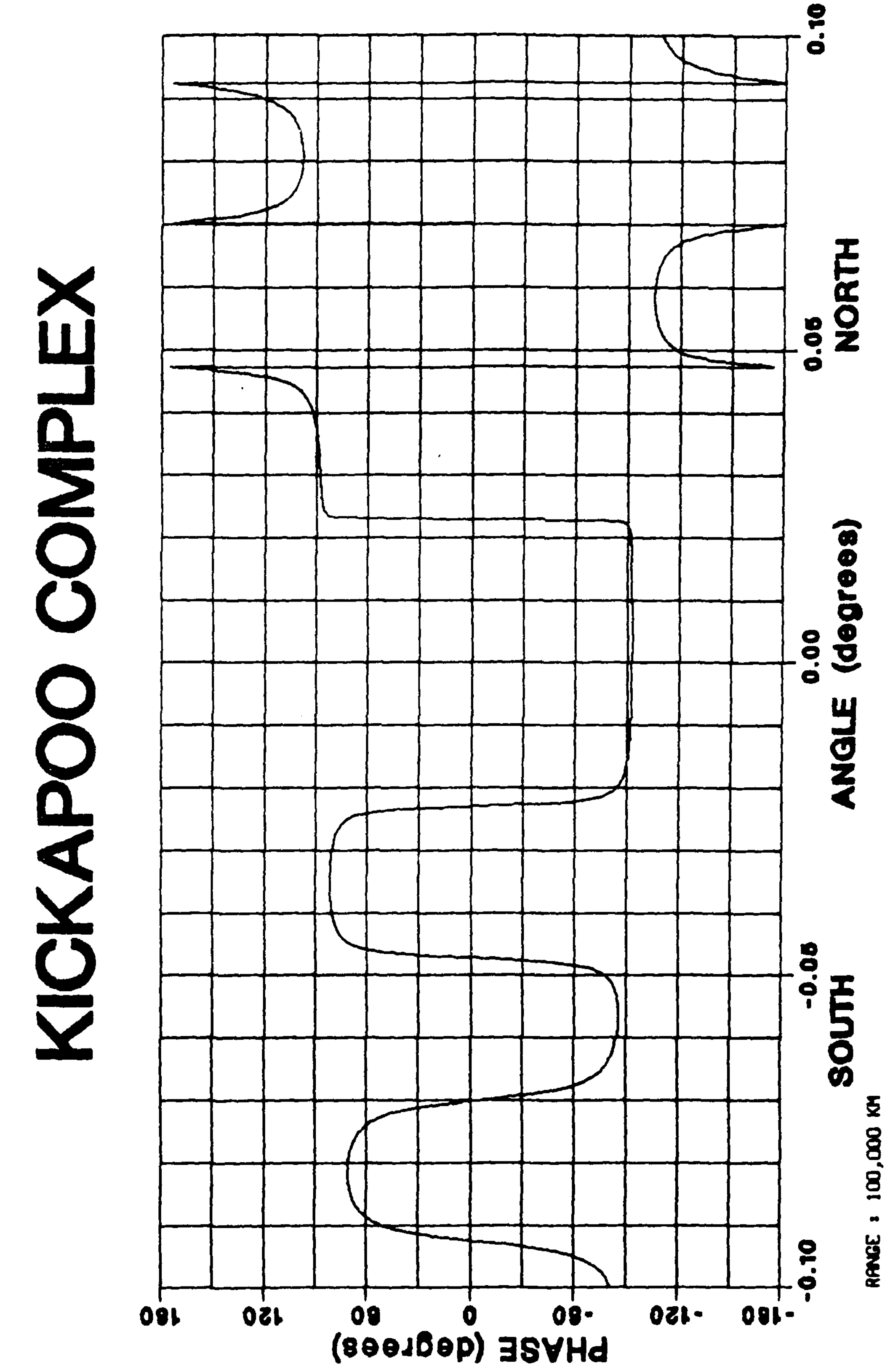

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.10
-0.05
0.00
0.05
0.10
SOUTH
ANGLE (degrees)
NORTH
RANGE : 100,000 KM

Figure 6b

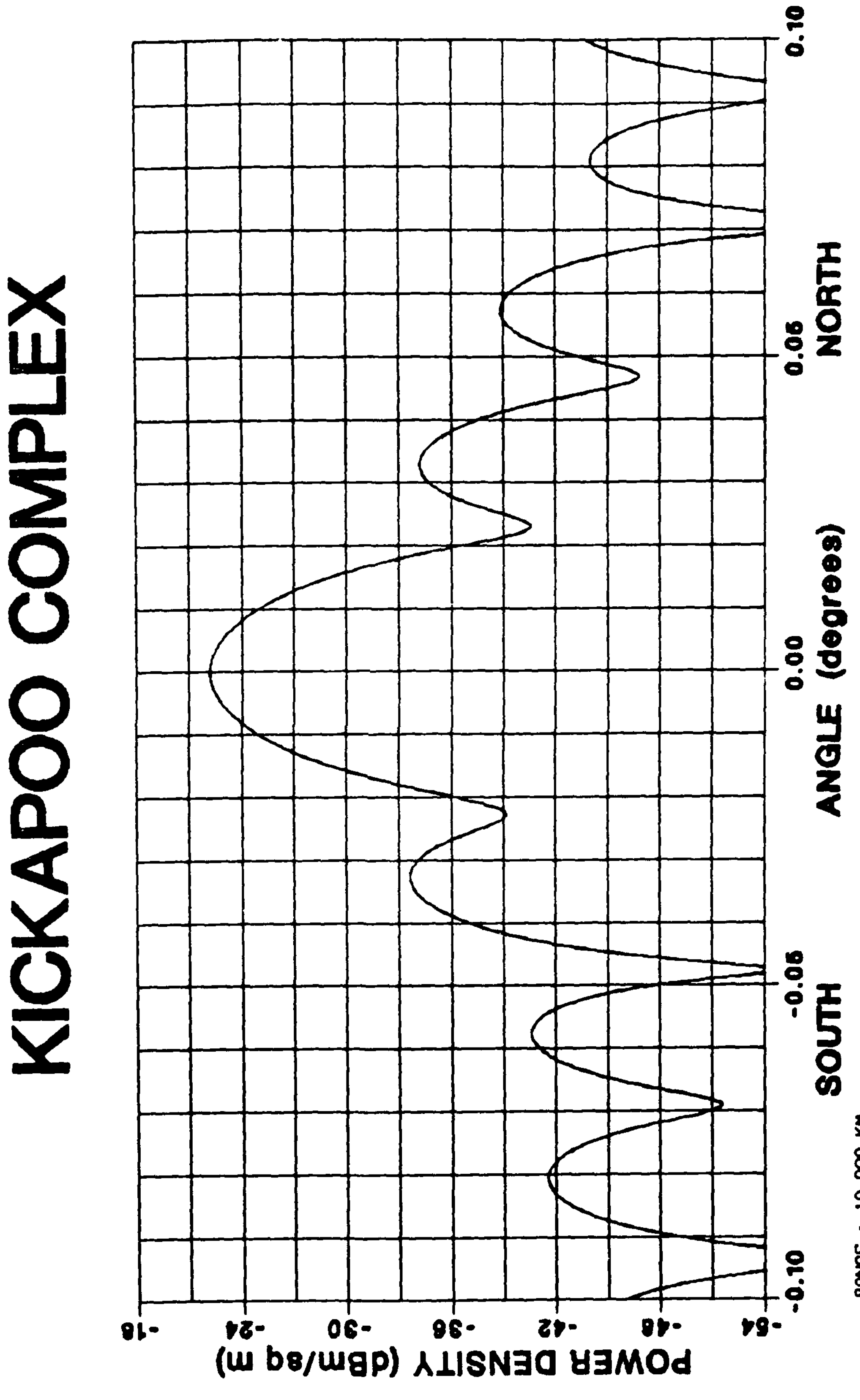

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-18
-24
-30
-36
-42
-48
-54
ANGLE (degrees)
SOUTH
NORTH
-0.10
-0.05
0.00
0.05
0.10
RANGE : 10,000 KM

Figure 7a

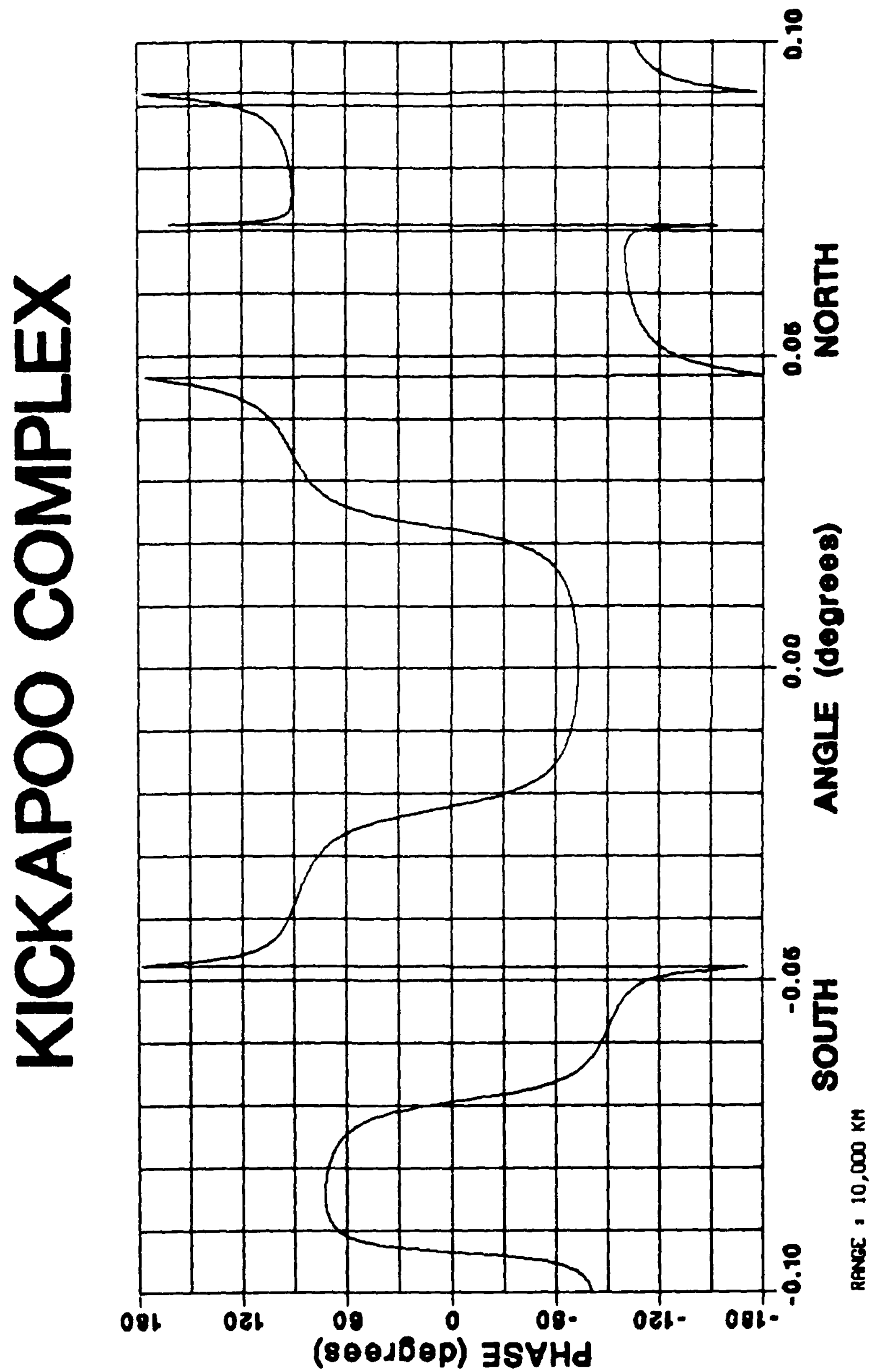

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.10
-0.05
0.00
0.05
0.10
SOUTH
ANGLE (degrees)
NORTH
RANGE : 10,000 KM

Figure 7b

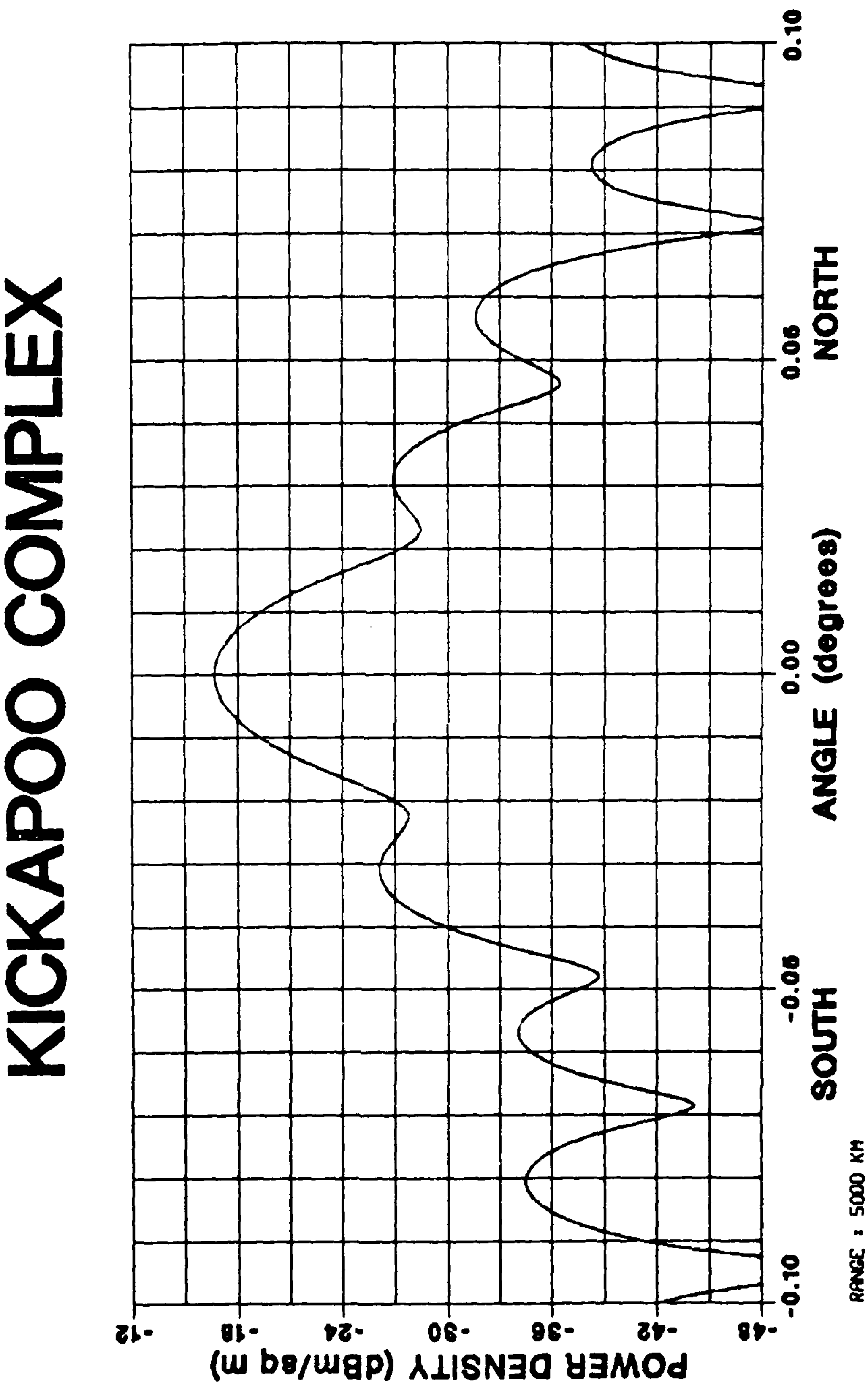
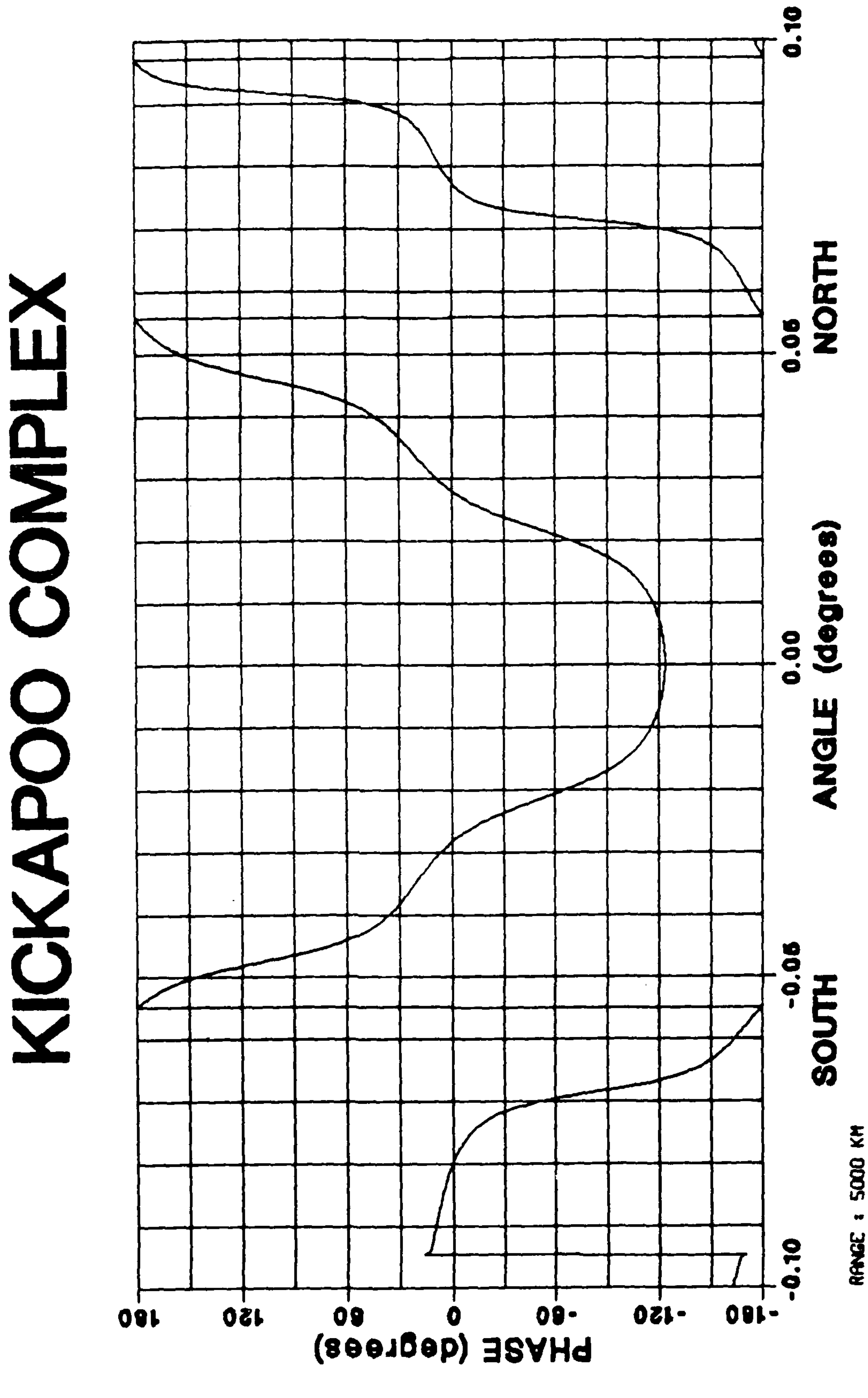

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-12
-18
-24
-30
-36
-42
-48
-0.10
-0.05
0.00
0.05
0.10
SOUTH
ANGLE (degrees)
NORTH
RANGE : 5000 KM

Figure 8a



KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.10
-0.05
0.00
0.05
0.10
SOUTH
ANGLE (degrees)
NORTH
RANGE : 5000 KM

Figure 8b

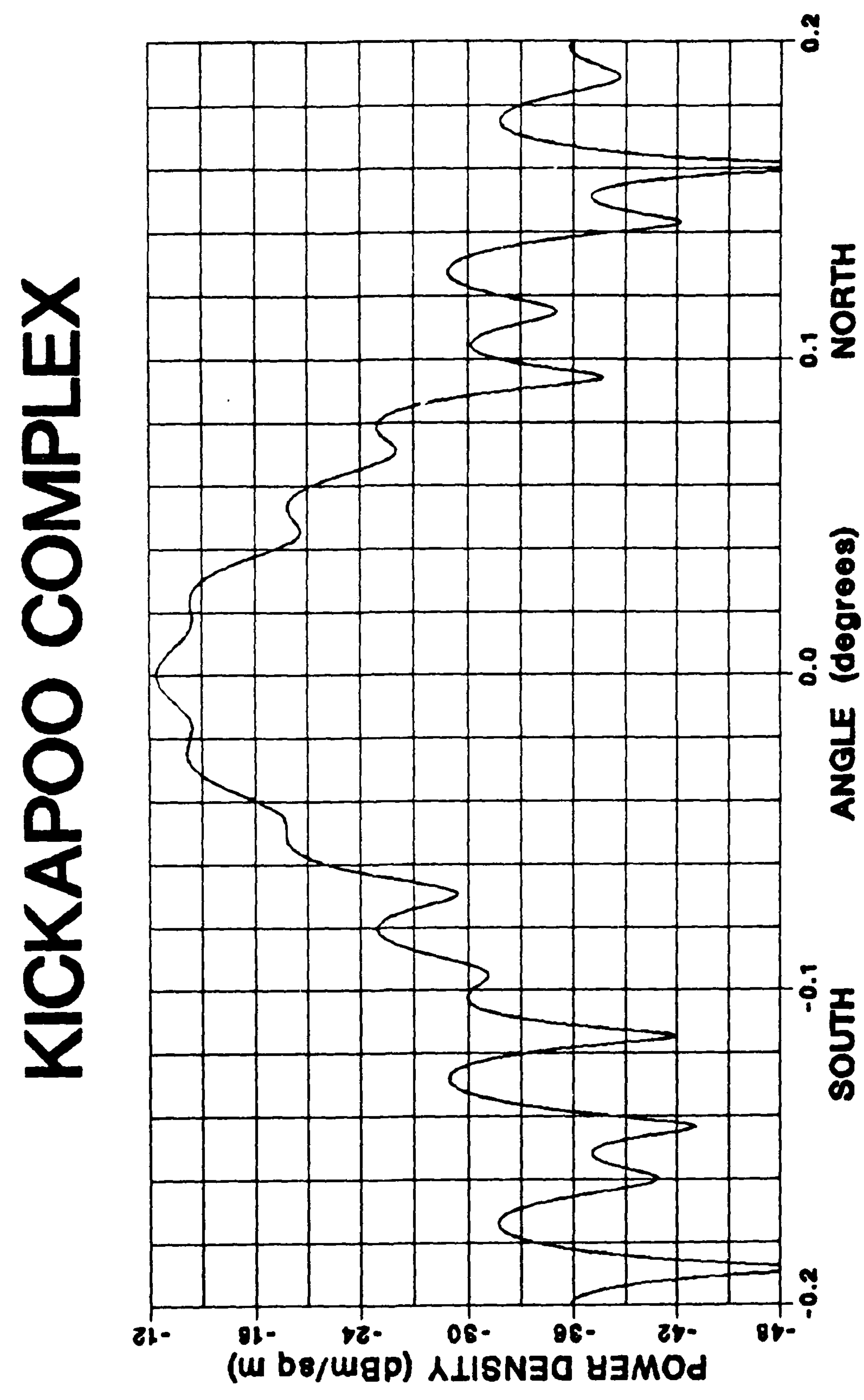

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-12
-18
-24
-30
-36
-42
-48
ANGLE (degrees)
-0.2
-0.1
0.0
0.1
0.2
SOUTH
NORTH
RANGE : 2000 KM

Figure 9a

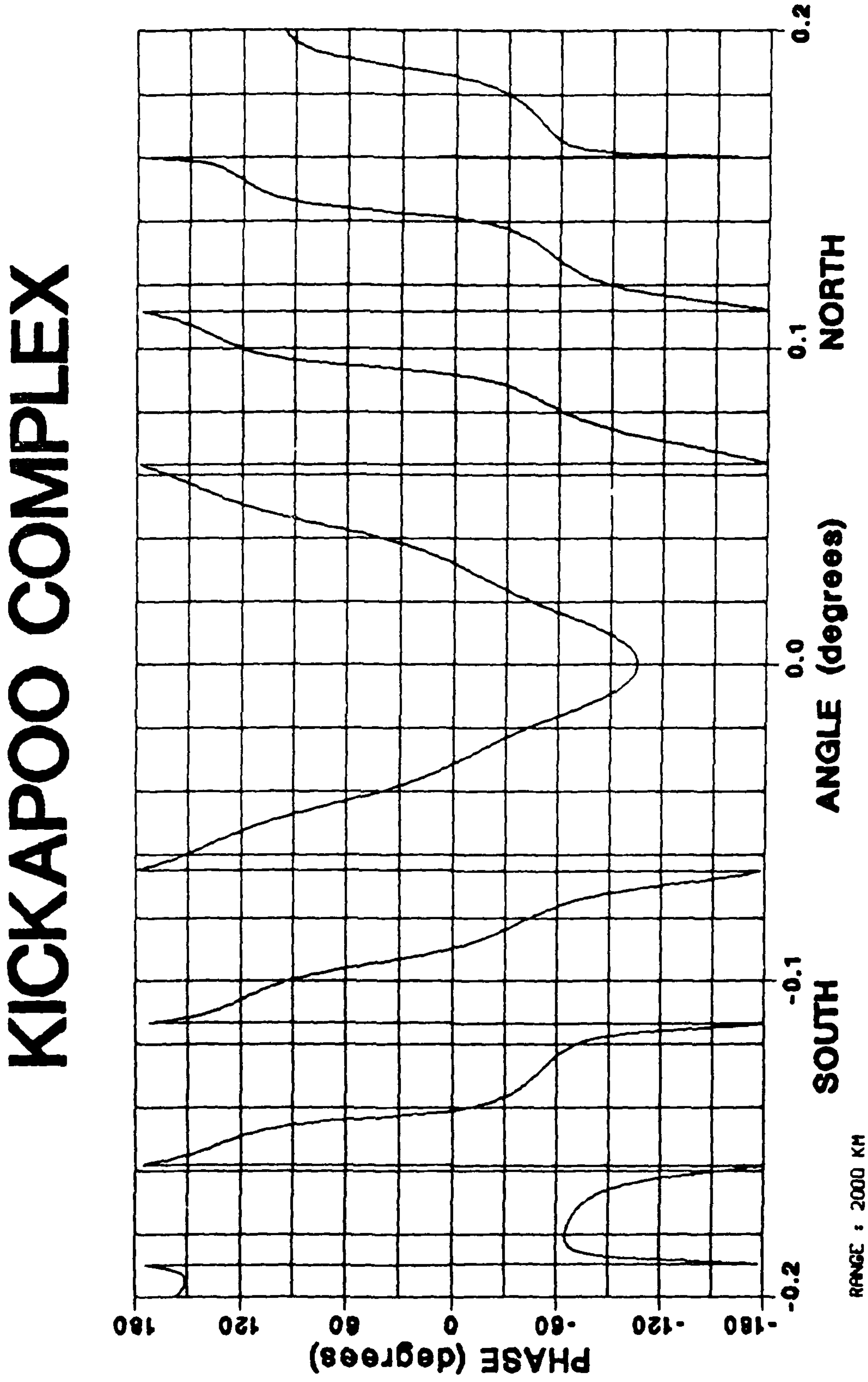

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.2
-0.1
0.0
0.1
0.2
SOUTH
ANGLE (degrees)
NORTH
RANGE : 2000 KM

Figure 9b

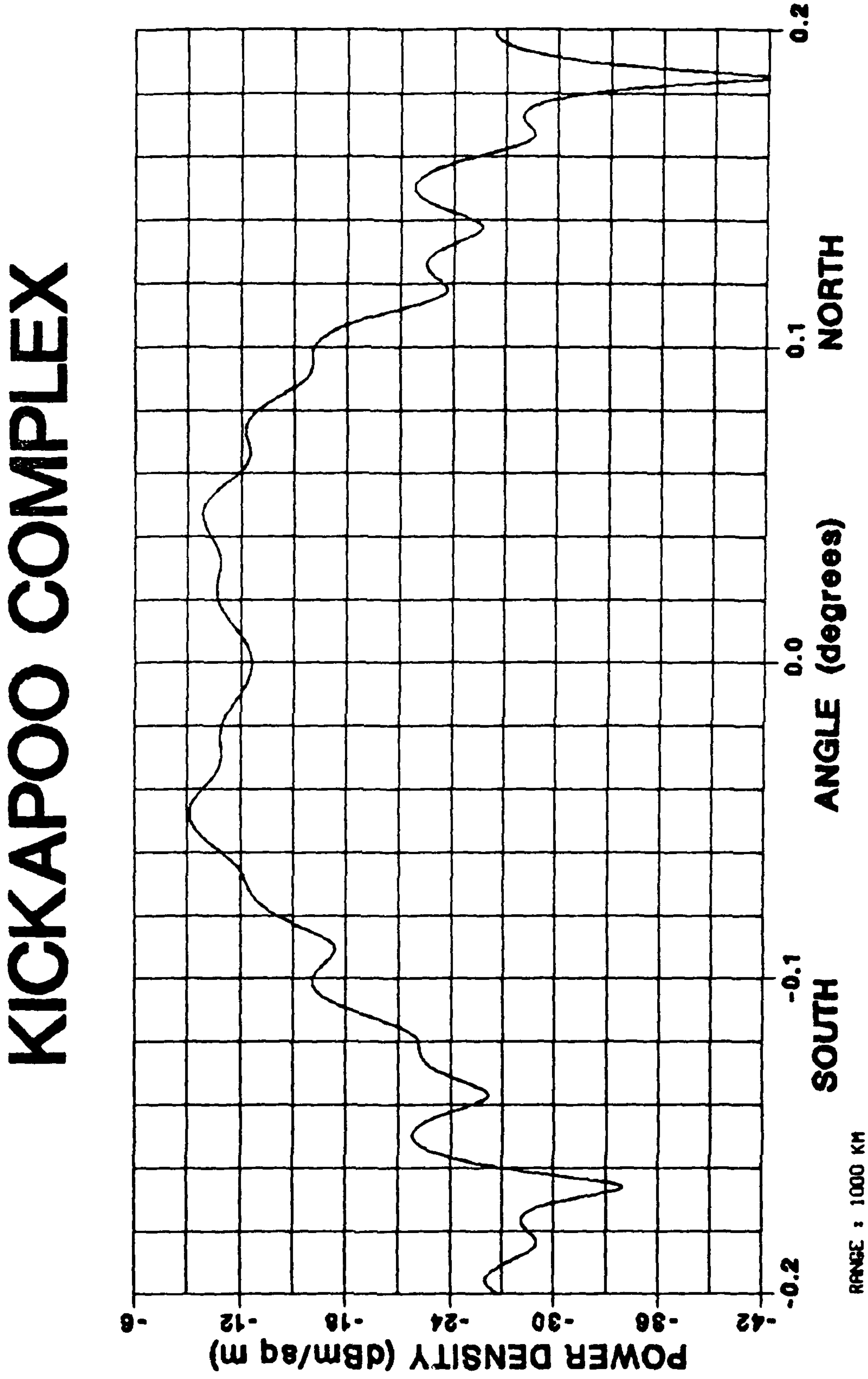

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-6
-12
-18
-24
-30
-36
-42
-0.2
-0.1
0.0
0.1
0.2
SOUTH
ANGLE (degrees)
NORTH
RANGE : 1000 KM

Figure 10a

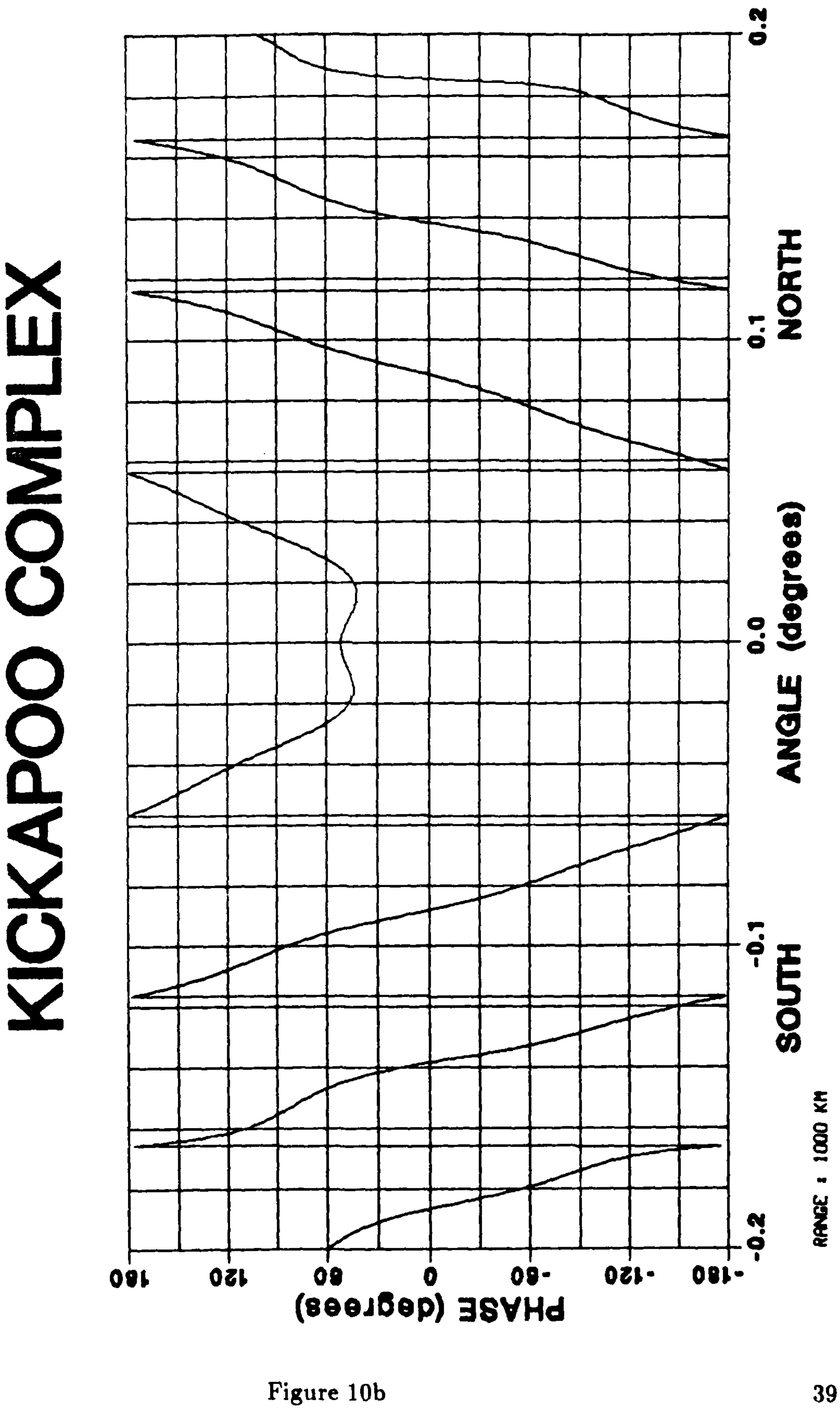

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.2
-0.1
0.0
0.1
0.2
SOUTH
ANGLE (degrees)
NORTH
RANGE : 1000 KM

Figure 10b

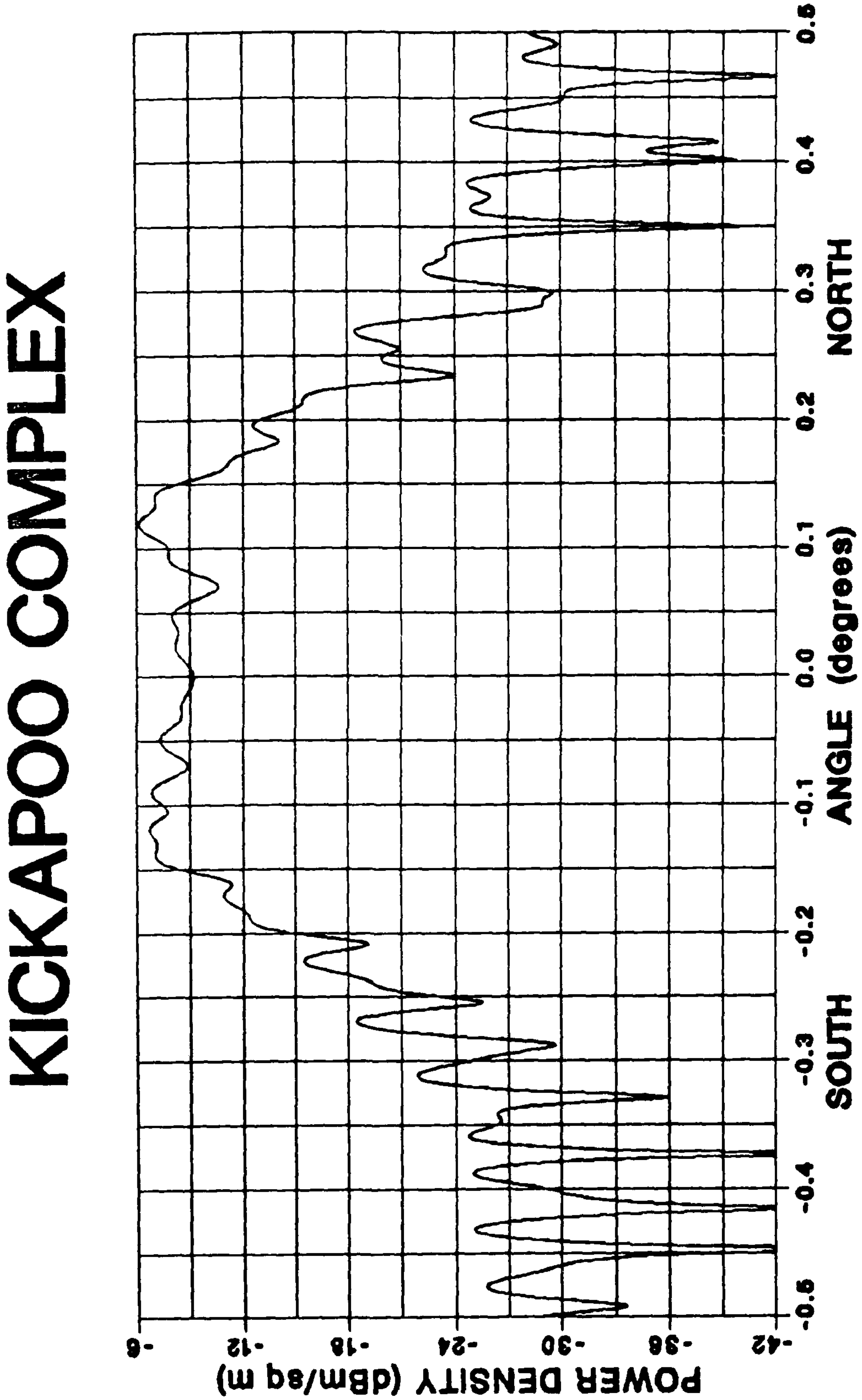

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-6
-12
-18
-24
-30
-36
-42
-0.5
-0.4
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
0.5
SOUTH
ANGLE (degrees)
NORTH
RANGE : 500 KM

Figure 11a

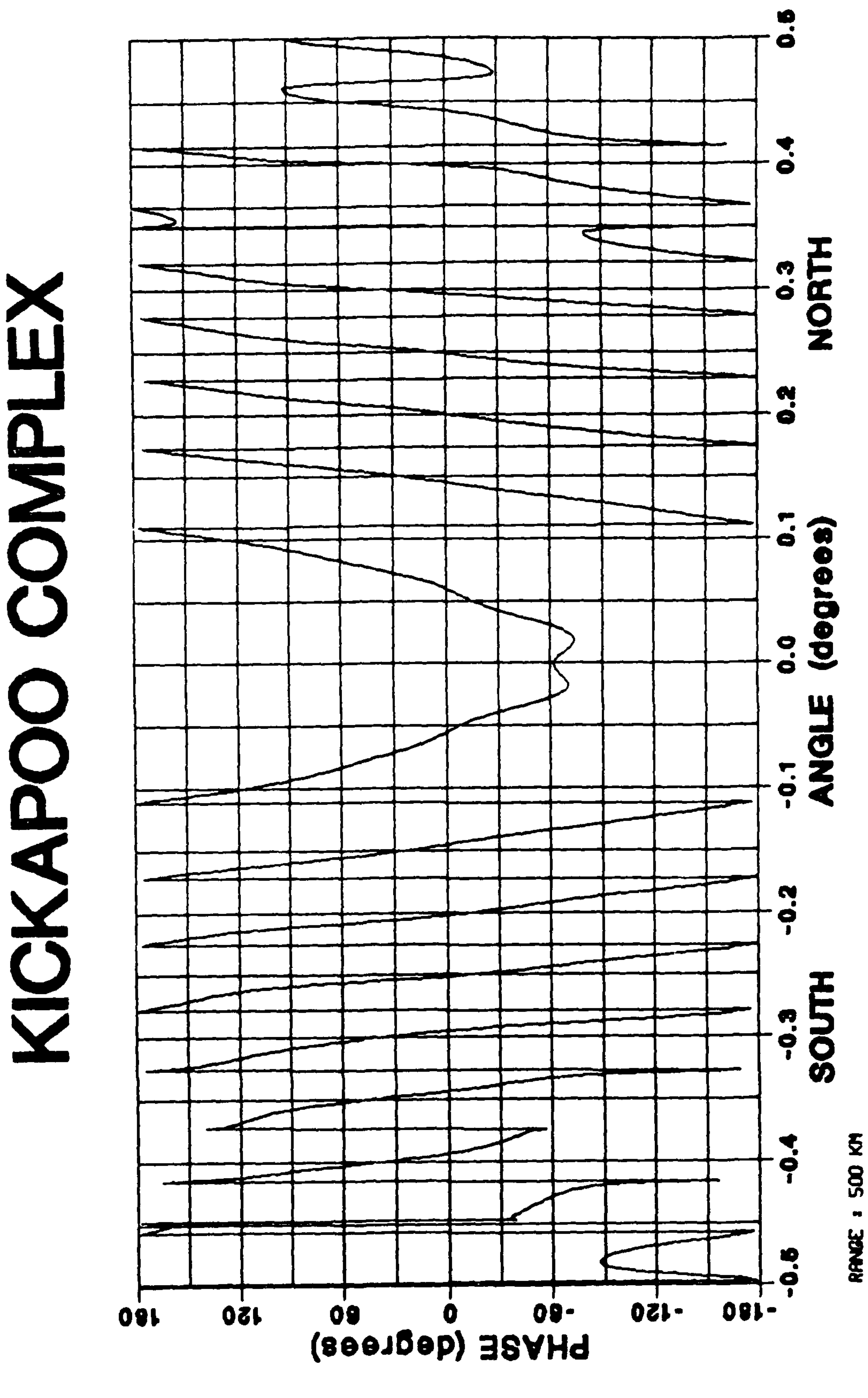

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
ANGLE (degrees)
SOUTH
NORTH
-0.5
-0.4
-0.3
-0.2
-0.1
0.0
0.1
0.2
0.3
0.4
0.5
RANGE : 500 KM

Figure 11b

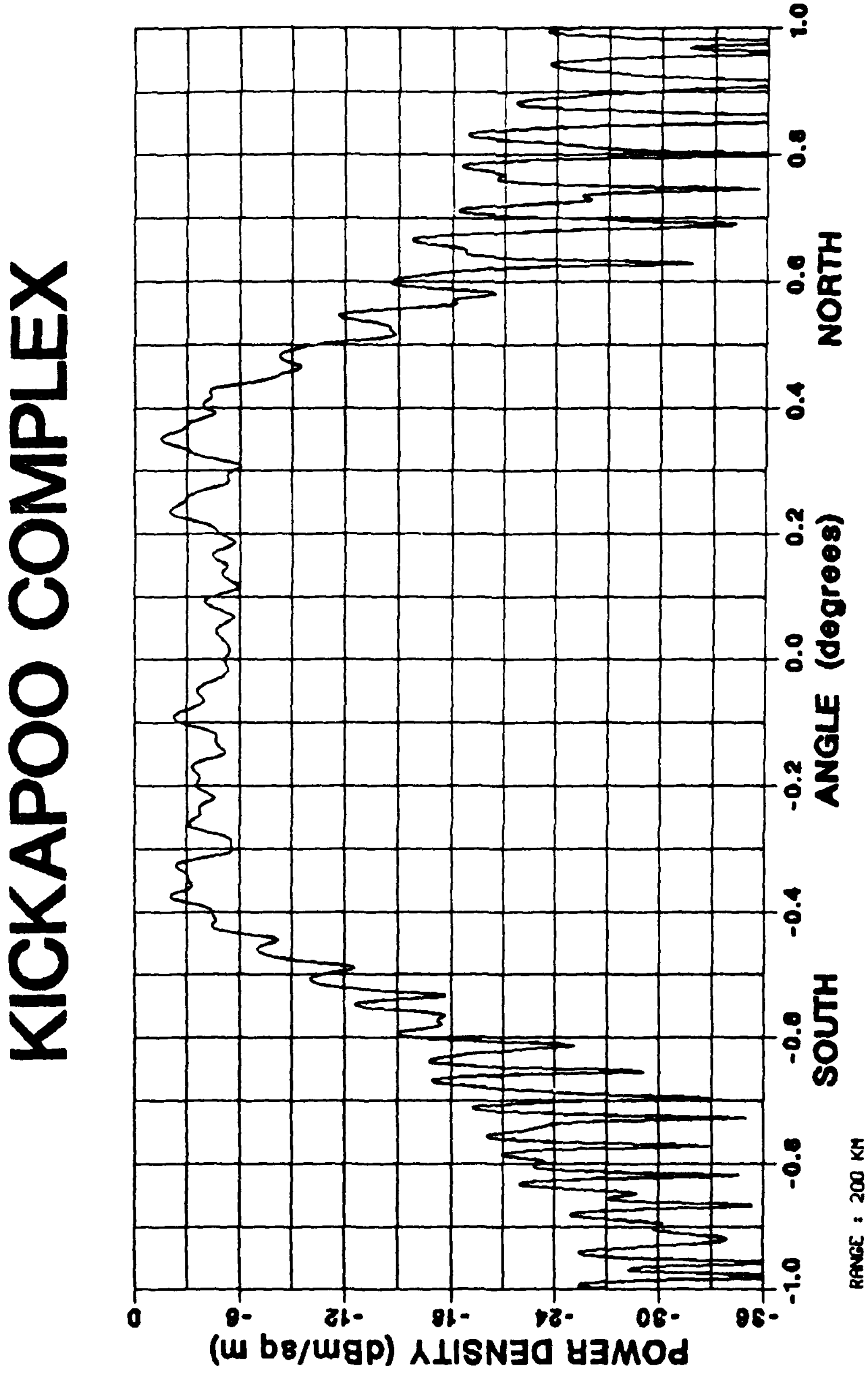

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
0
-6
-12
-18
-24
-30
-36
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
SOUTH
ANGLE (degrees)
NORTH
RANGE : 200 KM

Figure 12a

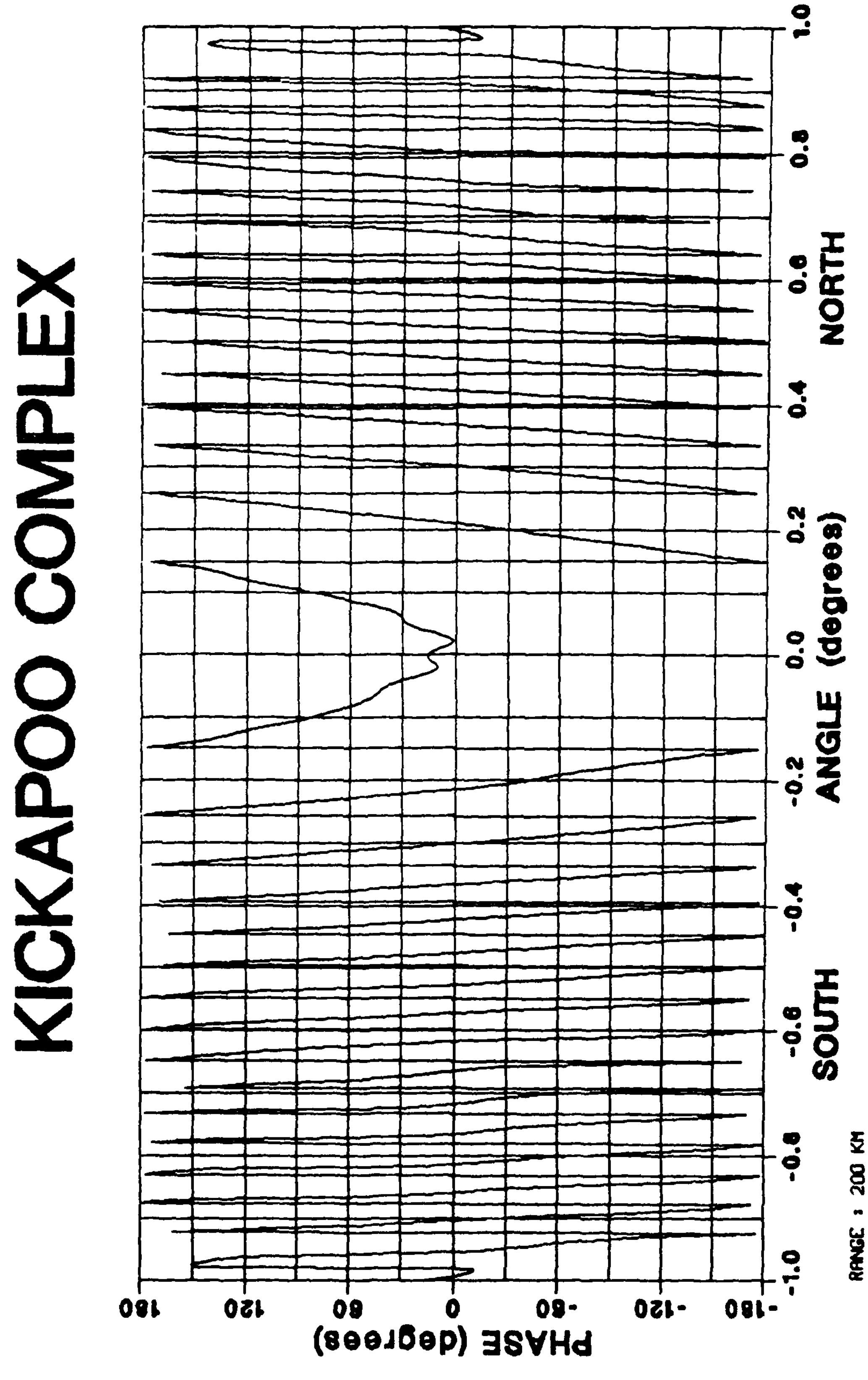
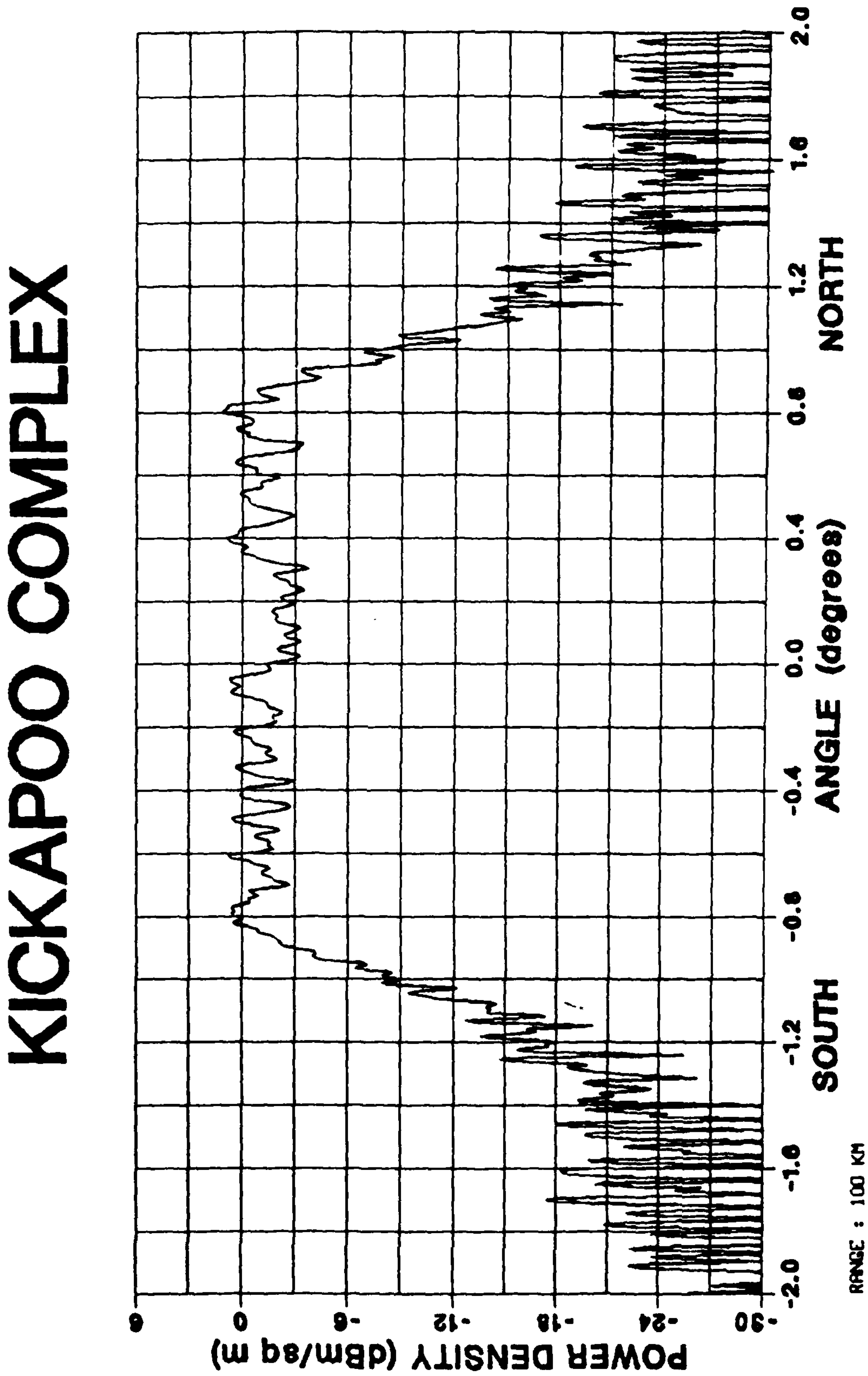

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
ANGLE (degrees)
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
SOUTH
NORTH
RANGE : 200 KM

Figure 12b



KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
6
0
-6
-12
-18
-24
-30
ANGLE (degrees)
-2.0
-1.6
-1.2
-0.8
-0.4
0.0
0.4
0.8
1.2
1.6
2.0
SOUTH
NORTH
RANGE : 100 KM

Figure 13a

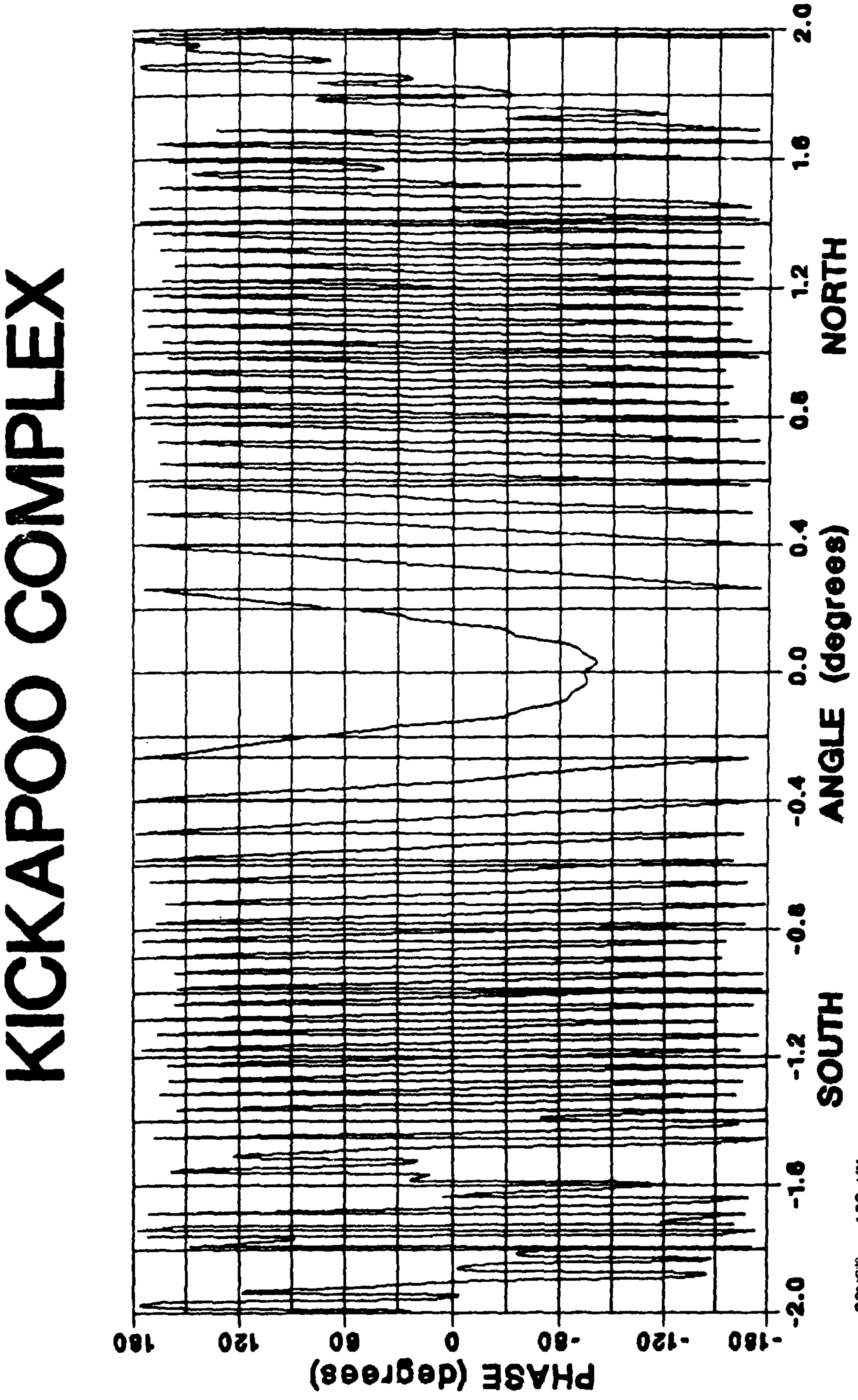

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
ANGLE (degrees)
SOUTH
NORTH
-2.0
-1.6
-1.2
-0.8
-0.4
0.0
0.4
0.8
1.2
1.6
2.0

Figure 13b

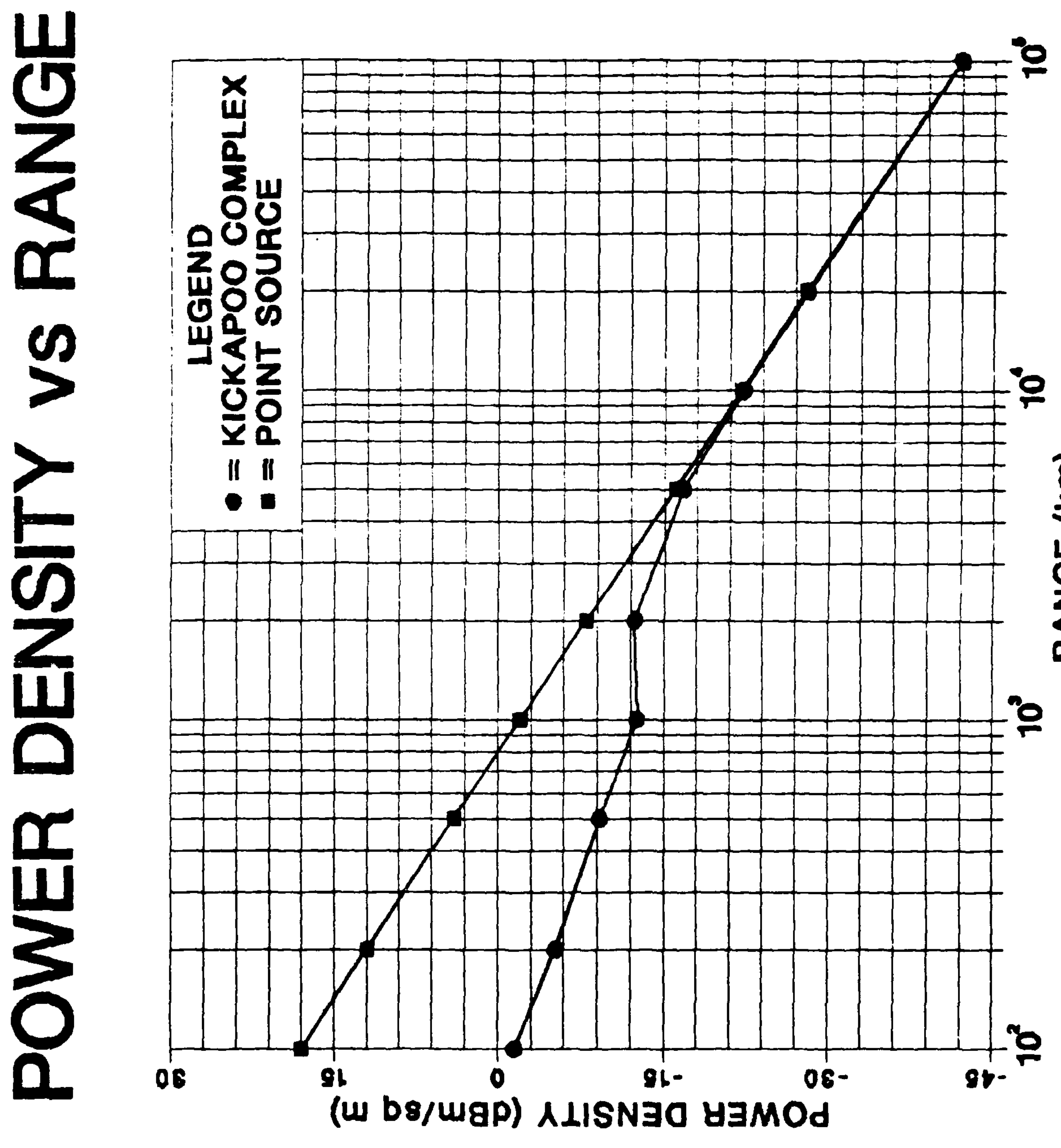

POWER DENSITY VS RANGE
LEGEND
● = KICKAPOO COMPLEX
■ = POINT SOURCE
POWER DENSITY (dBm/sq m)
30
15
0
-15
-30
-45
RANGE (km)
10^2
10^3
10^4
10^5

Figure 14

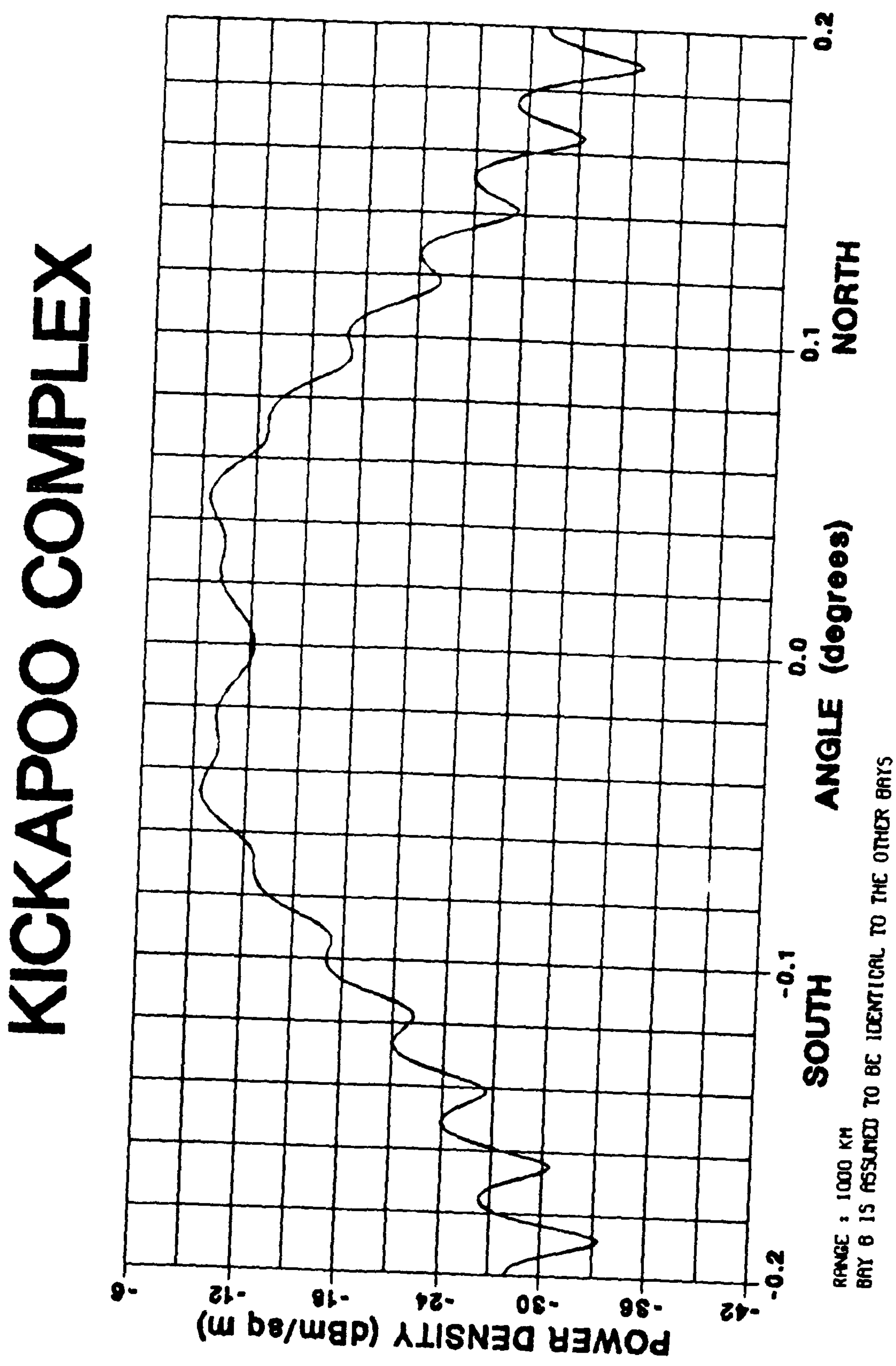

KICKAPOO COMPLEX
POWER DENSITY (dBm/sq m)
-6
-12
-18
-24
-30
-36
-42
-0.2
-0.1
0.0
0.1
0.2
SOUTH
ANGLE (degrees)
NORTH
RANGE : 1000 KM
BAY 6 IS ASSUMED TO BE IDENTICAL TO THE OTHER BAYS

Figure 15a

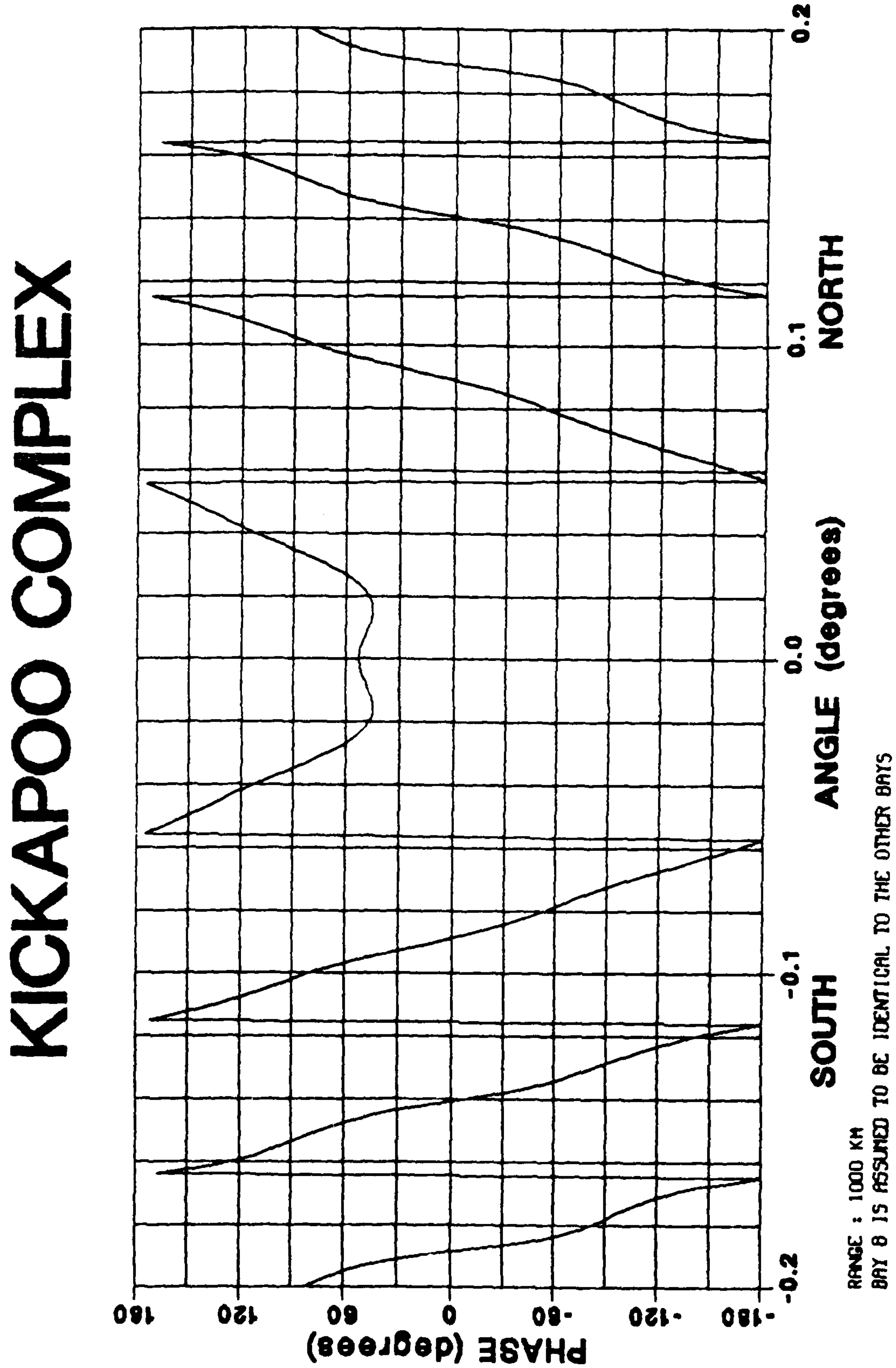

KICKAPOO COMPLEX
PHASE (degrees)
180
120
60
0
-60
-120
-180
-0.2
-0.1
0.0
0.1
0.2
SOUTH
ANGLE (degrees)
NORTH
RANGE : 1000 KM
BAY 8 IS ASSUMED TO BE IDENTICAL TO THE OTHER BAYS

Figure 15b

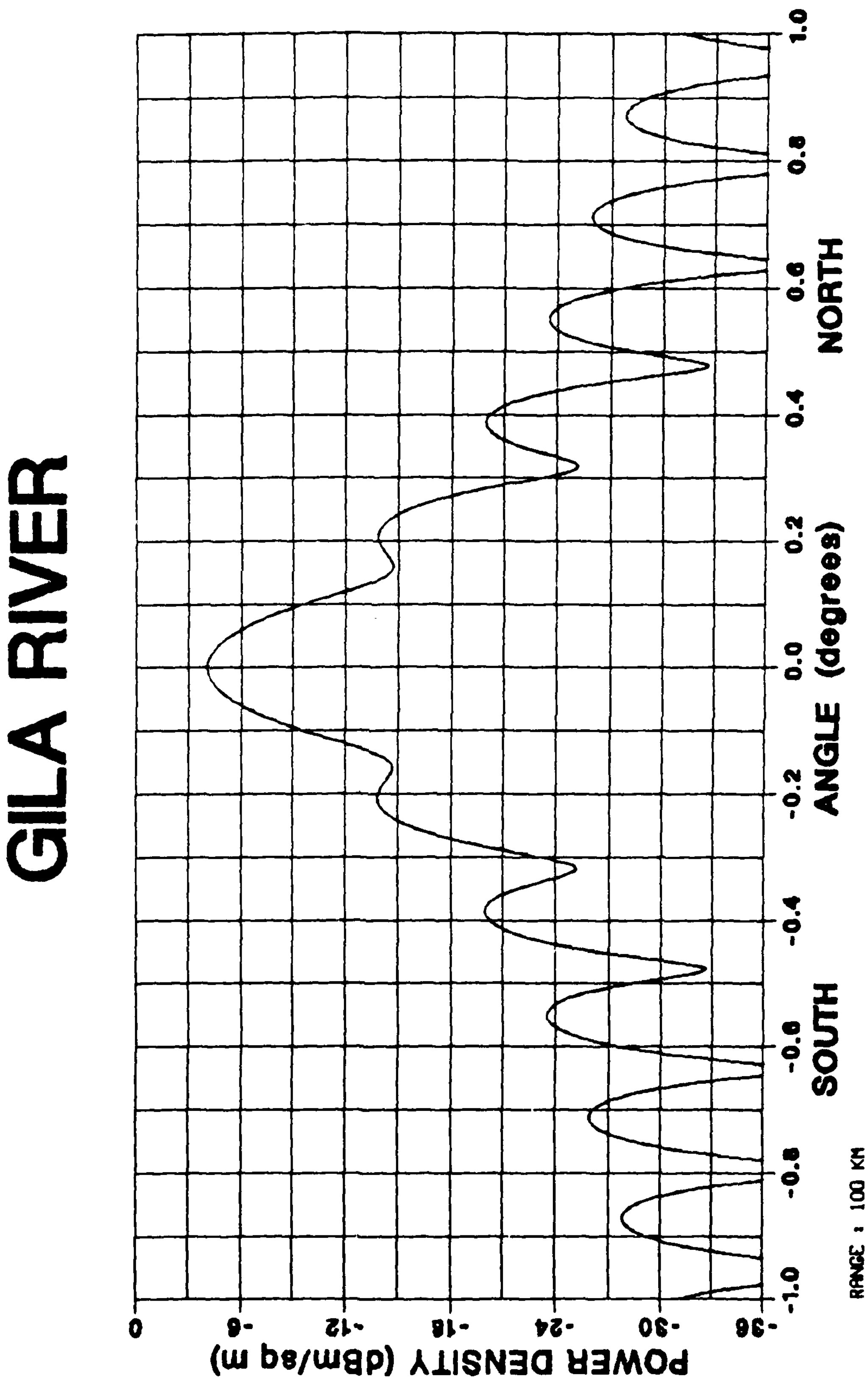

GILA RIVER
POWER DENSITY (dBm/sq m)
0
-6
-12
-18
-24
-30
-36
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
SOUTH
ANGLE (degrees)
NORTH
RANGE : 100 KM

Figure 16

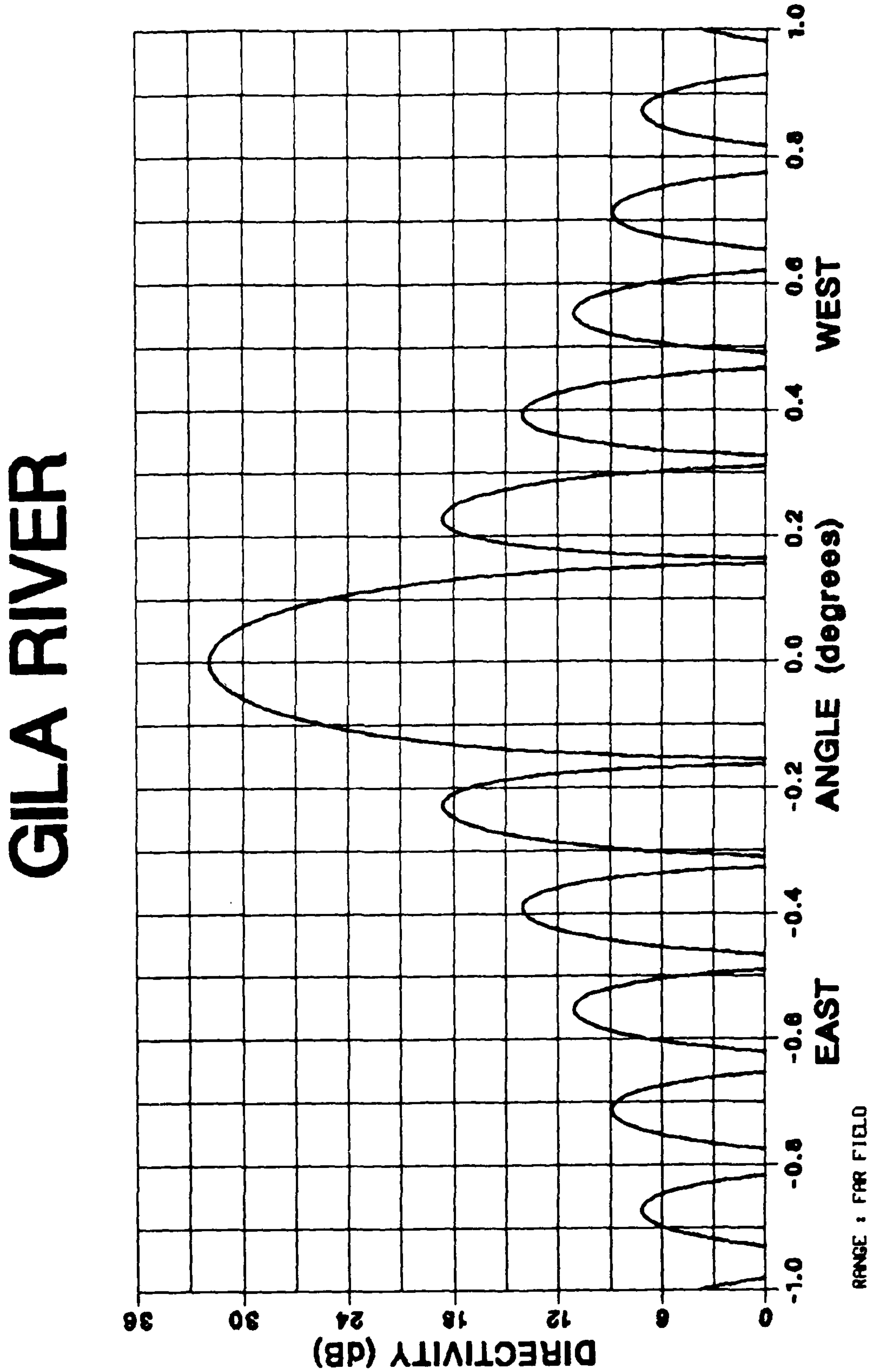

GILA RIVER
DIRECTIVITY (dB)
0
6
12
18
24
30
36
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
ANGLE (degrees)
EAST
WEST
RANGE : FAR FIELD

Figure 17

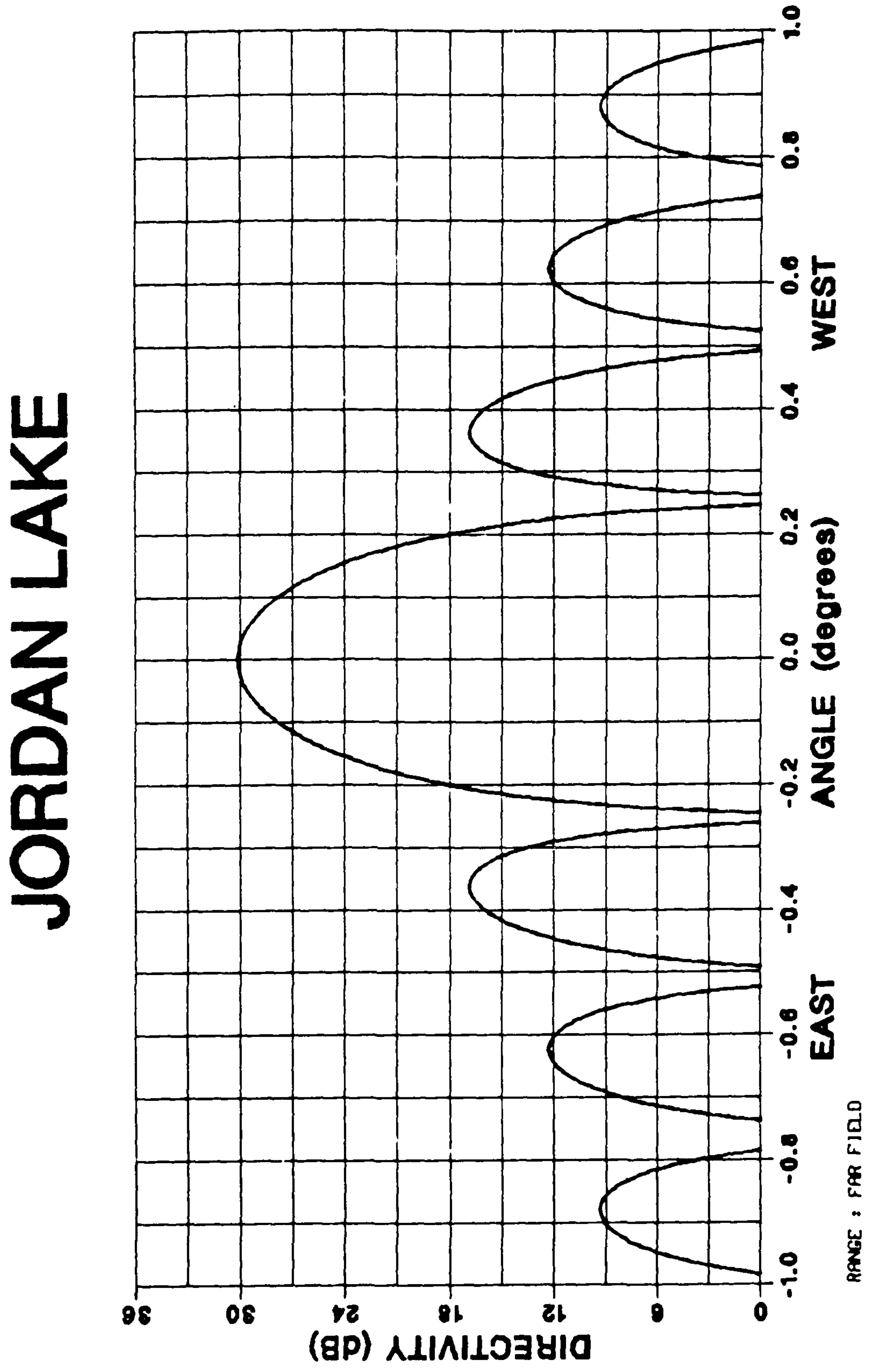

JORDAN LAKE
DIRECTIVITY (dB)
0
6
12
18
24
30
36
-1.0
-0.8
-0.6
-0.4
-0.2
0.0
0.2
0.4
0.6
0.8
1.0
EAST
ANGLE (degrees)
WEST
RANGE : FAR FIELD

Figure 18

```
C                CONDUCTING MEDIUM", National Technical
C                Information Service, Springfield, VA 22151,
C                NASA Contract Report CR-2399, June 1974.
C
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  Calculate mutual impedance.
C
C       2.  RETURN
C
C  INPUTS   EXPLICIT:
C       D       -
C       CGDS    -
C       CPSI    -
C       ETA     -
C       GAM     -
C       S1      -
C       S2      -
C       SGD1    -
C       SGD2    -
C       T1      -
C       T2      -
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       P11     -
C       P12     -
C       P21     -
C       P22     -
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C
C  SYSTEM STATE CHANGES:   NONE
C
C-
C
C  MODIFIED:
C       Dr. Steven L. Berg, 11/30/88
C               - Added header
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CGDS,SGDS,SGDT,P11,P12,P21,P22,CST,SGD1,SGD2,C,B
      COMPLEX*16 ET1,ET2,ES1,ES2,EXPA,EXPB,EB,EC,EK,EL,EKL,EGZI
      COMPLEX*16 E(2,2),F(2,2),EGZ(2,2),GM(2),GP(2)
      COMPLEX*16 ETA,GAM
      DATA PI/3.141592653589793/
      DSQ=D*D
      SGDS=SGD1
      IF(S2.LT.S1)SGDS=-SGD1
      SGDT=SGD2
      IF(T2.LT.T1)SGDT=-SGD2
      IF(DABS(CPSI).GT..997) GO TO 110
      ES1=CDEXP(GAM*S1)
      ES2=CDEXP(GAM*S2)
      ET1=CDEXP(GAM*T1)
```

```
      ET2=CDEXP(GAM*T2)
      DD=D
      DPSI=CPSI
      TD1=T1
      TD2=T2
      CPSS=DPSI*DPSI
      CD=DD/DSQRT(1.D0-CPSS)
      C=DCMPLX(0.0D0,CD)
      BD=CD*DPSI
      B=DCMPLX(0.0D0,BD)
      EB=CDEXP(GAM*B)
      EC=CDEXP(GAM*C)
      DO 10 K=1,2
       DO 10 L=1,2
 10   E(K,L)=(.0,.0)
      TS1=TD1*TD1
      TS2=TD2*TD2
      DPQ=DD*DD
      SI=S1
      DO 100 I=1,2
       FI=(-1)**I
       SDI=SI
       SIS=SDI*SDI
       ST1=2.*SDI*TD1*DPSI
       ST2=2.*SDI*TD2*DPSI
       R1=DSQRT(DPQ+SIS+TS1-ST1)
       R2=DSQRT(DPQ+SIS+TS2-ST2)
       EK=EB
       DO 50 K=1,2
        FK=(-1)**K
        SK=FK*SDI
        EL=EC
        DO 40 L=1,2
         FL=(-1)**L
         EKL=EK*EL
         XX=FK*BD+FL*CD
         TL1=FL*TD1
         TL2=FL*TD2
         RR1=R1+SK+TL1
         RR2=R2+SK+TL2
         CALL EXPJ(GAM*DCMPLX(RR1,-XX),GAM*DCMPLX(RR2,-XX),EXPA)
         CALL EXPJ(GAM*DCMPLX(RR1,XX),GAM*DCMPLX(RR2,XX),EXPB)
         E(K,L)=E(K,L)+FI*(EXPA*EKL+EXPB/EKL)
 40     EL=1./EC
 50    EK=1./EB
       ZD=SDI*DPSI
       ZC=ZD
       EGZI=CDEXP(GAM*ZC)
       RR1=R1+ZD-TD1
       RR2=R2+ZD-TD2
       CALL EXPJ(GAM*RR1,GAM*RR2,EXPB)
       RR1=R1-ZD+TD1
       RR2=R2-ZD+TD2
       CALL EXPJ(GAM*RR1,GAM*RR2,EXPA)
       F(I,1)=2.*SGDS*EXPA/EGZI
       F(I,2)=2.*SGDS*EXPB*EGZI
 100  SI=S2
      CST=ETA/(16.*PI*SGDS*SGDT)
      P11=CST*(( F(1,1)+E(2,2)*ES2-E(1,2)/ES2)*ET2
     2        +(-F(1,2)-E(2,1)*ES2+E(1,1)/ES2)/ET2)
```

```
      P12=CST*((-F(1,1)-E(2,2)*ES2+E(1,2)/ES2)*ET1
     2         +( F(1,2)+E(2,1)*ES2-E(1,1)/ES2)/ET1)
      P21=CST*((-F(2,1)-E(2,2)*ES1+E(1,2)/ES1)*ET2
     2         +( F(2,2)+E(2,1)*ES1-E(1,1)/ES1)/ET2)
      P22=CST*(( F(2,1)+E(2,2)*ES1-E(1,2)/ES1)*ET1
     2         +(-F(2,2)-E(2,1)*ES1+E(1,1)/ES1)/ET1)
      RETURN
  110 IF(CPSI.LT.0.)GO TO 120
      TA=T1
      TB=T2
      GO TO 130
  120 TA=-T1
      TB=-T2
      SGDT=-SGDT
  130 SI=S1
      DO 150 I=1,2
       TJ=TA
       DO 140 J=1,2
        ZIJ=TJ-SI
        R=DSQRT(DSQ+ZIJ*ZIJ)
        W=R+ZIJ
        IF(ZIJ.LT.0.)W=DSQ/(R-ZIJ)
        V=R-ZIJ
        IF(ZIJ.GT.0.)V=DSQ/(R+ZIJ)
        IF(J.EQ.1)V1=V
        IF(J.EQ.1)W1=W
        EGZ(I,J)=CDEXP(GAM*ZIJ)
  140  TJ=TB
       CALL EXPJ(GAM*V1,GAM*V,GP(I))
       CALL EXPJ(GAM*W1,GAM*W,GM(I))
  150 SI=S2
      CST=-ETA/(8.*PI*SGDS*SGDT)
      P11=CST*(GM(2)*EGZ(2,2)+GP(2)/EGZ(2,2)
     2-CGDS*(GM(1)*EGZ(1,2)+GP(1)/EGZ(1,2)))
      P12=CST*(-GM(2)*EGZ(2,1)-GP(2)/EGZ(2,1)
     2+CGDS*(GM(1)*EGZ(1,1)+GP(1)/EGZ(1,1)))
      P21=CST*(GM(1)*EGZ(1,2)+GP(1)/EGZ(1,2)
     2-CGDS*(GM(2)*EGZ(2,2)+GP(2)/EGZ(2,2)))
      P22=CST*(-GM(1)*EGZ(1,1)-GP(1)/EGZ(1,1)
     2+CGDS*(GM(2)*EGZ(2,1)+GP(2)/EGZ(2,1)))
      RETURN
      END
C
C**********************************************************************
C+
      SUBROUTINE EXPJ(V1,V2,W12)
C
C *********************************************************************
C *                                                                   *
C *    SUBROUTINE EXPJ                                                *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
C                            Information Service, Springfield, VA 22151,
C                            NASA Contract Report CR-2399, June 1974
C  DATE:        JUNE-1974
```

```
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED: NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This program calculates an exponential integral. For more
C  information consult APPENDIX 7 in
C               J. H. Richmond, "COMPUTER PROGRAM FOR
C               THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C               CONDUCTING MEDIUM", National Technical
C               Information Service, Springfield, VA 22151,
C               NASA Contract Report CR-2399, June 1974.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  Calculate exponential integral.
C
C       2.  RETURN
C
C  INPUTS   EXPLICIT:
C       V1      - lower limit of exponential integral
C       V2      - upper limit of exponential integral
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       W12     - exponential integral value
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C       Dr. Steven L. Berg, 11/30/88
C               - Added header
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 V1,V2,Z,E15,UC,VC,W12,S,EC,T
      DIMENSION V(21),W(21),D(16),E(16)
      DATA PI/3.141592653589793/
      DATA V/ 0.22284667E 00,
     20.11889321E 01,0.29927363E 01,0.57751436E 01,0.98374674E 01,
     30.15982874E 02,0.93307812E-01,0.49269174E 00,0.12155954E 01,
     40.22699495E 01,0.36676227E 01,0.54253366E 01,0.75659162E 01,
     50.10120228E 02,0.13130282E 02,0.16654408E 02,0.20776479E 02,
     60.25623894E 02,0.31407519E 02,0.38530683E 02,0.48026086E 02/
      DATA W/ 0.45896460E 00,
```

```
     20.41700083E 00,0.11337338E 00,0.10399197E-01,0.26101720E-03,
     30.89854791E-06,0.21823487E 00,0.34221017E 00,0.26302758E 00,
     40.12642582E 00,0.40206865E-01,0.85638778E-02,0.12124361E-02,
     50.11167440E-03,0.64599267E-05,0.22263169E-06,0.42274304E-08,
     60.39218973E-10,0.14565152E-12,0.14830270E-15,0.16005949E-19/
      DATA D/ 0.22495842E 02,
     2 0.74411568E 02,-0.41431576E 03,-0.78754339E 02, 0.11254744E 02,
     3 0.16021761E 03,-0.23862195E 03,-0.50094687E 03,-0.68487854E 02,
     4 0.12254778E 02,-0.10161976E 02,-0.47219591E 01, 0.79729681E 01,
     5-0.21069574E 02, 0.22046490E 01, 0.89728244E 01/
      DATA E/ 0.21103107E 02,
     2-0.37959787E 03,-0.97489220E 02, 0.12900672E 03, 0.17949226E 02,
     3-0.12910931E 03,-0.55705574E 03, 0.13524801E 02, 0.14696721E 03,
     4 0.17949528E 02,-0.32981014E 00, 0.31028836E 02, 0.81657657E 01,
     5 0.22236961E 02, 0.39124892E 02, 0.81636799E 01/
      Z=V1
      DO 100 JIM=1,2
       X=DREAL(Z)
       Y=DIMAG(Z)
       E15=(.0,.0)
       AB=CDABS(Z)
       IF(AB.EQ.0.) GO TO 90
       IF(X.GE.0. .AND. AB.GT.10.) GO TO 80
       YA=DABS(Y)
       IF(X.LE.0. .AND. YA.GT.10.) GO TO 80
       IF(YA-X.GE.17.5.OR.YA.GE.6.5.OR.X+YA.GE.5.5.OR.X.GE.3.) GO TO 20
       IF(X.LE.-9) GO TO 40
       IF(YA-X.GE.2.5) GO TO 50
       IF(X+YA.GE.1.5) GO TO 30
10     N=6.+3.*AB
       E15=1./(N-1.)-Z/N**2
15     N=N-1
       E15=1./(N-1.)-Z*E15/N
       IF(N.GE.3) GO TO 15
       E15=Z*E15-DCMPLX(.577216+DLOG(AB),DATAN2(Y,X))
       GO TO 90
20     J1=1
       J2=6
       GO TO 31
30     J1=7
       J2=21
31     S=(.0,.0)
       YS=Y*Y
       DO 32 I=J1,J2
        XI=V(I)+X
        CF=W(I)/(XI*XI+YS)
32     S=S+DCMPLX(XI*CF,-YA*CF)
       GO TO 54
40     T3=X*X-Y*Y
       T4=2.*X*YA
       T5=X*T3-YA*T4
       T6=X*T4+YA*T3
       UC=DCMPLX(D(11)+D(12)*X+D(13)*T3+T5-E(12)*YA-E(13)*T4,
     2          E(11)+E(12)*X+E(13)*T3+T6+D(12)*YA+D(13)*T4)
       VC=DCMPLX(D(14)+D(15)*X+D(16)*T3+T5-E(15)*YA-E(16)*T4,
     2          E(14)+E(15)*X+E(16)*T3+T6+D(15)*YA+D(16)*T4)
       GO TO 52
50     T3=X*X-Y*Y
       T4=2.*X*YA
       T5=X*T3-YA*T4
```

```
      T6=X*T4+YA*T3
      T7=X*T5-YA*T6
      T8=X*T6+YA*T5
      T9=X*T7-YA*T8
      T10=X*T8+YA*T7
      UC=DCMPLX(D(1)+D(2)*X+D(3)*T3+D(4)*T5+D(5)*T7+T9-(E(2)*YA+E(3)*T4
    2 +E(4)*T6+E(5)*T8),E(1)+E(2)*X+E(3)*T3+E(4)*T5+E(5)*T7+T10+
    3 (D(2)*YA+D(3)*T4+D(4)*T6+D(5)*T8))
      VC=DCMPLX(D(6)+D(7)*X+D(8)*T3+D(9)*T5+D(10)*T7+T9-(E(7)*YA+E(8)
    2 *T4+E(9)*T6+E(10)*T8),E(6)+E(7)*X+E(8)*T3+E(9)*T5+E(10)*T7+T10+
    3 (D(7)*YA+D(8)*T4+D(9)*T6+D(10)*T8))
 52   EC=UC/VC
      S=EC/DCMPLX(X,YA)
 54   EX=DEXP(-X)
      T=EX*DCMPLX(DCOS(YA),-DSIN(YA))
      E15=S*T
 56   IF(Y.LT.0.)E15=DCONJG(E15)
      GO TO 90
 80   E15=.409319/(Z+.193044)+.421831/(Z+1.02666)+.147126/(Z+2.56788)+
    2 .206335E-1/(Z+4.90035)+.107401E-2/(Z+8.18215)+.158654E-4/(Z+
    3 12.7342)+.317031E-7/(Z+19.3957)
      E15=E15*CDEXP(-Z)
 90   IF(JIM.EQ.1)W12=E15
 100  Z=V2
      Z=V2/V1
      TH=DATAN2(DIMAG(Z),DREAL(Z))-DATAN2(DIMAG(V2),DREAL(V2))
     2+DATAN2(DIMAG(V1),DREAL(V1))
      AB=DABS(TH)
      IF(AB.LT.1.)TH=.0
      IF(TH.GT.1.)TH=2.*PI
      IF(TH.LT.-1.)TH=-2.*PI
      THO=.0
      W12=W12-E15+DCMPLX(THO,TH)
      RETURN
      END
C
C**********************************************************************
C+
      SUBROUTINE VLTMTX(N,N1,I1,I2,I3,X,Y,Z,ETA,GAM,WL,NVL,CR,CI,IER)
C
C *********************************************************************
C *                                                                   *
C *    SUBROUTINE VLTMTX                                              *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         08-FEBRUARY-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  QATR    - computes an approximation for a
C                                 function f(l) integrated along l from
C                                 LL to LU, where l is a linear
C                                 coordinate along an arbitrary line in
C                                 space.
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
```

```
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates an excitation voltage matrix for the
C  electric field from a NAVSPASUR transmitter element incident on a
C  thin-wire structure.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  Select integration parameters.
C
C       2.  DO
C                for all test dipoles
C
C                    Select integration limits for first monopole.
C
C                    CALL
C                        - real part of excitation voltage.
C
C                    CALL
C                        - imaginary part of excitation voltage.
C
C                    Select integration limits for second monopole.
C
C                    CALL
C                        - real part of excitation voltage.
C
C                    CALL
C                        - imaginary part of excitation voltage.
C
C                    Add two monopole values together for matrix value.
C
C           ENDDO
C
C       3. RETURN
C
C  INPUTS   EXPLICIT:
C       ETA     - intrinsic impedance of ambient medium (mks units)
C       GAM     - complex propagation constant of ambient medium (mks
C                 units)
C       I1(I)   - 1st endpoint of test dipole I
C       I2(I)   - 2nd endpoint of test dipole I
C       I3(I)   - 3rd endpoint of test dipole I
C       N       - total # of test dipoles
C       N1      - # of test dipole modes in one quadrant of the ground
C                 screen, which have symmetric counter parts in the
C                 other three quadrants.
C       NVL     - column number for excitation voltage column matrix
C       X(I)    - x coordinate of endpoint I (meters)
C       Y(I)    - y coordinate of endpoint I (meters)
C       Z(I)    - z coordinate of endpoint I (meters)
C       WL      - wavelength (meters)
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       CI(I,NVL)       - imaginary part of excitation voltage matrix
```

```
C                          for and electric file incident on the
C                          thin-wire structure.
C       CR(I,NVL)        - real part of excitation voltage matrix for and
C                          electric file incident on the thin-wire
C                          structure.
C       IER      - error indicator for numerical integration
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       AR       - length of test monopole
C       EPS      - upper limit of absolute error
C       FCT1     - name of external function subprogram which computes
C                  the value of the tangential component of the incident
C                  NAVSPASUR transmitter dipole electric field at a
C                  selected point along the test monopole weighted by the
C                  first term of the test dipole current distribution
C       FCT2     - name of external function subprogram which computes
C                  the value of the tangential component of the incident
C                  NAVSPASUR transmitter dipole electric field at a
C                  selected point along the test monopole weighted by the
C                  second term of the test dipole current distribution
C       NDIM     - the dimension of the auxiliary storage array AUX.
C                  NDIM-1 is the maximal number of bisections of the
C                  integration interval
C       RO       - length of test monopole projection onto x-y plane
C       XL       - x coordinate of lower limit of line integral
C       XU       - x coordinate of upper limit of line integral
C       YL       - y coordinate of lower limit of line integral
C       YU       - y coordinate of upper limit of line integral
C       YY1I     - imaginary part of line integral of FCT1 over monopole
C       YY1R     - real part of line integral of FCT1 over monopole
C       YY2I     - imaginary part of line integral of FCT2 over monopole
C       YY2R     - real part of line integral of FCT2 over monopole
C       ZL       - z coordinate of lower limit of line integral
C       ZU       - z coordinate of upper limit of line integral
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C
C  Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,GAM
      DIMENSION CR(602,602),CI(602,602)
      DIMENSION X(1),Y(1),Z(1),I1(1),I2(1),I3(1)
C
C  Declare functions.
C
      EXTERNAL FCT1,FCT2
C
C  Define integration parameters.
C
      EPS=1.D-4
      NDIM=100
C
C  Fill excitation voltage matrix with value for each test dipole.
```

```
C
      DO 10 I=1,N
       II=I
       IF(I.GT.N1) II=I+3*N1
C
C  Determine length and projected length of first test monopole.
C
       J=I1(II)
       K=I2(II)
       RO=DSQRT((X(K)-X(J))**2+(Y(K)-Y(J))**2)
       AR=DSQRT(RO**2+(Z(K)-Z(J))**2)
C
C  Determine integration limits.
C
       XL=X(J)
       XU=X(K)
       YL=Y(J)
       YU=Y(K)
       ZL=Z(J)
       ZU=Z(K)
C
C  Determine real part of excitation voltage value for first monopole.
C
       CALL QATR(XL,XU,YL,YU,ZL,ZU,EPS,NDIM,ETA,GAM,WL,RO,AR,FCT1,
     2 YY1R,IER,0)
       IF(IER.NE.0.) GO TO 30
C
C  Determine imaginary part of excitation voltage value for first
C  monopole.
C
       CALL QATR(XL,XU,YL,YU,ZL,ZU,EPS,NDIM,ETA,GAM,WL,RO,AR,FCT1,
     2 YY1I,IER,1)
       IF(IER.NE.0.) GO TO 30
C
C  Determine length and projected length of second test monopole.
C
       J=I2(II)
       K=I3(II)
       RO=DSQRT((X(K)-X(J))**2+(Y(K)-Y(J))**2)
       AR=DSQRT(RO**2+(Z(K)-Z(J))**2)
C
C  Determine integration limits.
C
       XL=X(J)
       XU=X(K)
       YL=Y(J)
       YU=Y(K)
       ZL=Z(J)
       ZU=Z(K)
C
C  Determine real part of excitation voltage value for second monopole.
C
       CALL QATR(XL,XU,YL,YU,ZL,ZU,EPS,NDIM,ETA,GAM,WL,RO,AR,FCT2,
     2 YY2R,IER,0)
       IF(IER.NE.0.) GO TO 30
C
C  Determine imaginary part of excitation voltage value for second
C  monopole.
C
       CALL QATR(XL,XU,YL,YU,ZL,ZU,EPS,NDIM,ETA,GAM,WL,RO,AR,FCT2,
```

```
     2 YY2I,IER,1)
       IF(IER.NE.0.) GO TO 30
C
C  Add two parts together.
C
       CR(I,NVL)=YY1R+YY2R
       CI(I,NVL)=YY1I+YY2I
 10   CONTINUE
C
C Return to main program
C
 30   RETURN
      END
C
C*********************************************************************
C+
      FUNCTION FCT1(X,Y,Z,XL,YL,ZL,XU,YU,ZU,ETA,GAM,WL,RO,AR,IRI)
C
C ********************************************************************
C *                                                                  *
C *     FUNCTION FCT1                                                *
C *                                                                  *
C ********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         08-FEBRUARY-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This function computes the value of the tangential component of
C  the incident NAVSPASUR transmitter dipole electric field at a
C  selected point along the test monopole weighted by the first term of
C  the test dipole current distribution.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Determine complex current value at point of interest.
C
C       2. Determine tangential electric field componenent at point of
C          interest.
C
C       3. Obtain product of tangential electric field value and complex
C          current value.
C
C       4. Select imaginary or real part of product.
C
C       5. RETURN
C
C  INPUTS   EXPLICIT:
```

```
C       AR      - length of test monopole
C       ETA     - intrinsic impedance of ambient medium (mks units)
C       GAM     - complex propagation constant of ambient medium (mks
C                 units)
C       IRI     - indicator for real or imaginary part of function value
C                       0 => real part
C                       1 => imaginary part
C       RO      - length of test monopole projection onto x-y plane
C       WL      - wavelength (meters)
C       X       - x coordinate at selected point along monopole
C       XL      - x coordinate of lower limit of test monopole
C       XU      - x coordinate of upper limit of test monopole
C       Y       - y coordinate at selected point along monopole
C       YL      - y coordinate of lower limit of test monopole
C       YU      - y coordinate of upper limit of test monopole
C       Z       - z coordinate at selected point along monopole
C       ZL      - z coordinate of lower limit of test monopole
C       ZU      - z coordinate of upper limit of test monopole
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       FCT1    - the value of the tangential component of the incident
C                 NAVSPASUR transmitter dipole electric field at a
C                 selected point along the test monopole weighted by the
C                 first term of the test dipole current distribution
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       FAC     - test dipole current value
C       CTF     - complex value of tangential component of electric
C                 field from NAVSPASUR transmitter element
C
C  SYSTEM STATE CHANGES: NONE
C
C-
C
C  MODIFIED:
C
C  Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CDIM
      COMPLEX*16 ETA,GAM,ZZ2,EGD,SGDT,SGDB,CEXPRP,CEXPR,CFR,CFT,CFP,
     2CR,CT,CP,CX,CY,CZ,CTF,FAC
C
C  Define constants.
C
      DATA PI/3.141592653589793/
      TP=2*PI
C
C  Calculate complex current value at selected point.
C
      SPHI=(YU-YL)/RO
      CPHI=(XU-XL)/RO
      STHT=RO/AR
      CTHT=(ZU-ZL)/AR
      EL=DSQRT((X-XL)**2+(Y-YL)**2+(Z-ZL)**2)
      EGD=CDEXP(GAM*EL)
```

```
      SGDT=(EGD-1./EGD)/2.
      EGD=CDEXP(GAM*AR)
      SGDB=(EGD-1./EGD)/2.
      FAC=SGDT/SGDB
C
C Calculate incident tangential electric field at selected point.
C
      ZZ2=ETA/2.
      CR=DCMPLX(0.,0.)
      CT=DCMPLX(0.,0.)
      CP=DCMPLX(0.,0.)
      DO 275 M=1,2
      FM=(-1.)**M
      H=FM*0.25*WL
      THETO=PI/180.*(90.-FM*55.)
      CTO=DCOS(THETO)
      STO=DSIN(THETO)
      RH=DSQRT((X+H*STO)**2+Y*Y+(Z+H*CTO)**2)
      CDIM=DCMPLX(0.,1.)
      PFD=DSQRT(X*X+Y*Y)
      RFD=DSQRT(PFD*PFD+Z*Z)
      CST=Z/RFD
      SNT=PFD/RFD
      CSP=1.0
      SNP=0.0
      IF(PFD.NE.0.)CSP=X/PFD
      IF(PFD.NE.0.)SNP=Y/PFD
      CEXPRP=CDEXP(-GAM*RH)/RH
      CEXPR=CDEXP(-GAM*RFD)
      CSTP=CTO*CST+STO*SNT*CSP
      SNTP=DSQRT(1-CSTP*CSTP)
      TNTP=SNTP/CSTP
      THFC=(CTO*SNT-STO*CST*CSP)/SNTP
      PHFC=(STO*SNP)/SNTP
      CFR=(H/RFD*CEXPRP-FM*CDIM*CEXPR/GAM/RFD/RFD)*ZZ2/TP*CDIM
      CFT=(CEXPRP*(1./SNTP+H/RFD/TNTP)+CEXPR/RFD/TNTP*FM*CDIM)*THFC
     2*ZZ2/TP*CDIM
      CFP=(CEXPRP*(1./SNTP+H/RFD/TNTP)+CEXPR/RFD/TNTP*FM*CDIM)*PHFC
     2*ZZ2/TP*CDIM
      CR=CR+CFR
      CT=CT+CFT
      CP=CP+CFP
 275  CONTINUE
      CX=CR*SNT*CSP+CT*CST*CSP-CP*SNP
      CY=CR*SNT*SNP+CT*CST*SNP+CP*CSP
      CZ=CR*CST-CT*SNT
      CTF=CX*STHT*CPHI+CY*STHT*SPHI+CZ*CTHT
C
C  Choose real or imaginary part of product of complex current and
C  tangential electric field.
C
      FCT1=DREAL(FAC*CTF)
      IF(IRI.NE.0) FCT1=DIMAG(FAC*CTF)
C
C  Return to subroutine.
C
      RETURN
      END
C
C***********************************************************************
```

```
C+
      FUNCTION FCT2(X,Y,Z,XL,YL,ZL,XU,YU,ZU,ETA,GAM,WL,RO,AR,IRI)
C
C ********************************************************************
C *                                                                  *
C *     FUNCTION FCT2                                                *
C *                                                                  *
C ********************************************************************
C  AUTHOR:        DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:          08-FEBRUARY-1988
C  LANGUAGE:      VAX FORTRAN
C  FILE:          MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This function computes the value of the tangential component of
C  the incident NAVSPASUR transmitter dipole electric field at a
C  selected point along the test monopole weighted by the second term of
C  the test dipole current distribution.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Determine complex current value at point of interest.
C
C       2. Determine tangential electric field componenent at point of
C          interest.
C
C       3. Obtain product of tangential electric field value and complex
C          current value.
C
C       4. Select imaginary or real part of product.
C
C       5. RETURN
C
C  INPUTS   EXPLICIT:
C       AR       - length of test monopole
C       ETA      - intrinsic impedance of ambient medium (mks units)
C       GAM      - complex propagation constant of ambient medium (mks
C                  units)
C       IRI      - indicator for real or imaginary part of function value
C                        0 => real part
C                        1 => imaginary part
C       RO       - length of test monopole projection onto x-y plane
C       WL       - wavelength (meters)
C       X        - x coordinate at selected point along monopole
C       XL       - x coordinate of lower limit of test monopole
C       XU       - x coordinate of upper limit of test monopole
C       Y        - y coordinate at selected point along monopole
C       YL       - y coordinate of lower limit of test monopole
C       YU       - y coordinate of upper limit of test monopole
C       Z        - z coordinate at selected point along monopole
```

```
C       ZL      - z coordinate of lower limit of test monopole
C       ZU      - z coordinate of upper limit of test monopole
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       FCT2    - the value of the tangential component of the incident
C                 NAVSPASUR transmitter dipole electric field at a
C                 selected point along the test monopole weighted by the
C                 second term of the test dipole current distribution
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       FAC     - test dipole current value
C       CTF     - complex value of tangential component of electric
C                 field from NAVSPASUR transmitter element
C
C  SYSTEM STATE CHANGES: NONE
C
C-
C
C  MODIFIED:
C
C  Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CDIM
      COMPLEX*16 ETA,GAM,ZZ2,EGD,SGDT,SGDB,CEXPRP,CEXPR,CFR,CFT,CFP,
     2CR,CT,CP,CX,CY,CZ,CTF,FAC
C
C  Define constants.
C
      DATA PI/3.141592653589793/
      TP=2*PI
C
C  Calculate complex current value at selected point.
C
      SPHI=(YU-YL)/RO
      CPHI=(XU-XL)/RO
      STHT=RO/AR
      CTHT=(ZU-ZL)/AR
      EL=DSQRT((XU-X)**2+(YU-Y)**2+(ZU-Z)**2)
      EGD=CDEXP(GAM*EL)
      SGDT=(EGD-1./EGD)/2.
      EGD=CDEXP(GAM*AR)
      SGDB=(EGD-1./EGD)/2.
      FAC=SGDT/SGDB
C
C Calculate incident tangential electric field at selected point.
C
      ZZ2=ETA/2.
      CR=DCMPLX(0.,0.)
      CT=DCMPLX(0.,0.)
      CP=DCMPLX(0.,0.)
      DO 275 M=1,2
      FM=(-1.)**M
      H=FM*0.25*WL
      THETO=PI/180.*(90.-FM*55.)
      CTO=DCOS(THETO)
```

```
      STO=DSIN(THETO)
      RH=DSQRT((X+H*STO)**2+Y*Y+(Z+H*CTO)**2)
      CDIM=DCMPLX(0.,1.)
      PFD=DSQRT(X*X+Y*Y)
      RFD=DSQRT(PFD*PFD+Z*Z)
      CST=Z/RFD
      SNT=PFD/RFD
      CSP=1.0
      SNP=0.0
      IF(PFD.NE.0.)CSP=X/PFD
      IF(PFD.NE.0.)SNP=Y/PFD
      CEXPRP=CDEXP(-GAM*RH)/RH
      CEXPR=CDEXP(-GAM*RFD)
      CSTP=CTO*CST+STO*SNT*CSP
      SNTP=DSQRT(1-CSTP*CSTP)
      TNTP=SNTP/CSTP
      THFC=(CTO*SNT-STO*CST*CSP)/SNTP
      PHFC=(STO*SNP)/SNTP
      CFR=(H/RFD*CEXPRP-FM*CDIM*CEXPR/GAM/RFD/RFD)*ZZ2/TP*CDIM
      CFT=(CEXPRP*(1./SNTP+H/RFD/TNTP)+CEXPR/RFD/TNTP*FM*CDIM)*THFC
     2*ZZ2/TP*CDIM
      CFP=(CEXPRP*(1./SNTP+H/RFD/TNTP)+CEXPR/RFD/TNTP*FM*CDIM)*PHFC
     2*ZZ2/TP*CDIM
      CR=CR+CFR
      CT=CT+CFT
      CP=CP+CFP
  275 CONTINUE
      CX=CR*SNT*CSP+CT*CST*CSP-CP*SNP
      CY=CR*SNT*SNP+CT*CST*SNP+CP*CSP
      CZ=CR*CST-CT*SNT
      CTF=CX*STHT*CPHI+CY*STHT*SPHI+CZ*CTHT
C
C  Choose real or imaginary part of product of complex current and
C  tangential electric field.
C
      FCT2=DREAL(FAC*CTF)
      IF(IRI.NE.0) FCT2=DIMAG(FAC*CTF)
C
C  Return to subroutine.
C
      RETURN
      END
C
C***********************************************************************
C+
      SUBROUTINE QATR(XL,XU,YL,YU,ZL,ZU,EPS,NDIM,ETA,GAM,WL,RO,AR,
     2FCT,YY,IER,IRI)
C
C **********************************************************************
C *                                                                    *
C *     SUBROUTINE QATR                                                *
C *                                                                    *
C **********************************************************************
C
C  AUTHOR:      DIGITAL EQUIPMENT CORPORATION
C               SOURCE: SUBROUTINE QUATR
C                       DEC SCIENTIFIC SUBROUTINES PACKAGE
C  DATE:        JUNE 1980
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770:: SPACE:[BERG.THINWIRE]THINWIRE.FOR
```

```
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine computes an approximation for a function f(l)
C  integrated along l from LL to LU, where l is a the linear coordinate
C  along an arbitrary line in space.
C       The evaluation of the integral is done by means of trapezoidal
C  rule in connection with Romberg's principle.  On return the
C  calculated value is the best possible approximation of the intergral
C  value and vector AUX, the upward diagonal of Romberg scheme.
C  Components AUX(I) (I=1,2,...,IEND, with IEND less than or equal to
C  NDIM) become approximations to integral value with decreasing
C  accuracy by multiplication with (LL-LU). For reference, see
C
C           (1) FILIPPI, DAS VERFAHREN VON ROMBERG-STIEFEL-BAUER ALS
C               SPEZIALFALL DES ALLGEMEINEN PRINZIPS VON RICHARDSON,
C               MATHEMATIK-TECHNIK-WIRTSCHAFT, VOL.11, ISS.2 (1964),
C               PP.49-54.
C           (2) BAUER, ALGORITHM 60, CACM, VOL.4, ISS.6 (1961), PP.255.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  Evaluate integral using trapezoidal rule in connection to
C           Romberg's principle.
C
C       2.  RETURN
C
C  INPUTS   EXPLICIT:
C       AR       - length of test monopole
C       EPS      - upper limit of absolute error
C       ETA      - intrinsic impedance of ambient medium (mks units)
C       FCT      - name of the external function subprogram used
C                  (requires an external statement)
C       GAM      - complex propagation constant of ambient medium (mks
C                  units)
C       IRI      - indicator for real or imaginary part of function value
C                        0 => real part
C                        1 => imaginary part
C       NDIM     - the dimension of the auxiliary storage array AUX
C                  NDIM-1 is the maximal number of bisections of the
C                  integration interval
C       RO       - length of test monopole projection onto x-y plane
C       WL       - wavelength (meters)
C       XL       - x coordinate of lower limit of line integral
C       XU       - x coordinate of upper limit of line integral
C       YL       - y coordinate of lower limit of line integral
C       YU       - y coordinate of upper limit of line integral
C       ZL       - z coordinate of lower limit of line integral
C       ZU       - z coordinate of upper limit of line integral
C
C           IMPLICIT:   NONE
```

```
C
C  OUTPUTS  EXPLICIT:
C       IER      - resulting error parameter
C                     0 => It was possible to reach the required accuracy;
C                          no error.
C                     1 => It was impossible to reach the required accuracy
C                          because of rounding errors.
C                     2 => It was impossible to check accuracy because NDIM
C                          is less than 5, of the required accuracy could
C                          not be reached within NDIM-1 steps; NDIM should
C                          be increased.
C       YY       - resulting approximation for the integral value
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       AUX      - an auxiliary storage array with dimension NDIM
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C       Dr. Steven L. BERG, 2/08/88
C                - Adapted to THINWIRE's requirements.
C       Dr. Steven L. BERG, 11/30/88
C                - Added header.
C
C  Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION AUX(100)
      COMPLEX*16 ETA,GAM
C
C  Preparations of Romberg-loop.
C
      AUX(1)=.5*(FCT(XL,YL,ZL,XL,YL,ZL,XU,YU,ZU,ETA,GAM,WL,RO,AR,
     2IRI)+FCT(XU,YU,ZU,XL,YL,ZL,XU,YU,ZU,ETA,GAM,WL,RO,AR,IRI))
      HX=XU-XL
      HY=YU-YL
      HZ=ZU-ZL
      H=AR
      IF(NDIM-1)8,8,1
    1 IF(H)2,10,2
C
C  NDIM is greater than 1 and H is not equal to 0.
C
    2 HHX=HX
      HHY=HY
      HHZ=HZ
      E=EPS/DABS(H)
      DELT2=0.
      P=1.
      JJ=1
      DO 7 I=2,NDIM
      YY=AUX(1)
      DELT1=DELT2
      HDX=HHX
      HDY=HHY
      HDZ=HHZ
```

```
      HHX=.5*HHX
      HHY=.5*HHY
      HHZ=.5*HHZ
      P=.5*P
      X=XL+HHX
      Y=YL+HHY
      Z=ZL+HHZ
      SM=0.
      DO 3 J=1,JJ
      SM=SM+FCT(X,Y,Z,XL,YL,ZL,XU,YU,ZU,ETA,GAM,WL,RO,AR,IRI)
      X=X+HDX
      Y=Y+HDY
 3    Z=Z+HDZ
      AUX(I)=.5*AUX(I-1)+P*SM
C
C  A new approximation of integral value is computed by means of
C  trapezoidal rule.
C
C  Start of Romberg's extrapolation method.
C
      Q=1.
      JI=I-1
      DO 4 J=1,JI
      II=I-J
      Q=Q+Q
      Q=Q+Q
    4 AUX(II)=AUX(II+1)+(AUX(II+1)-AUX(II))/(Q-1.)
C
C  End of Romberg-step.
C
      DELT2=DABS(YY-AUX(1))
      IF(I-8)7,5,5
    5 IF(DELT2-E)10,10,6
    6 IF(DELT2-DELT1)7,11,11
    7 JJ=JJ+JJ
    8 IER=2
    9 YY=H*AUX(1)
      RETURN
   10 IER=0
      GO TO 9
   11 IER=1
      YY=H*YY
C
C  Return to subroutine.
C
      RETURN
      END
C
C**********************************************************************
C+
      SUBROUTINE CROUT(N,NVL,CR,CI,LND)
C
C *********************************************************************
C *                                                                   *
C *     SUBROUTINE CROUT                                              *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:       G. A. THIELE, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C                REFERENCE :  G. A. Thiele, WIRE_GRID BODY PROGRAM from
```

```
C                              "WIRE ANTENNAS", Chapter 2 in Computer
C                              Techniques for Electromagnetics, Pergamon
C                              Press, New York, 1973
C  DATE:        1973
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine solves for an unknown complex column matrix [B],
C  from the matrix equation [A][B] = [C], where [A] is a known complex
C  square matrix and [C] is a known complex column matrix.  [A], [B],
C  and [C] are each stored in the form of two real matrices, one
C  containing the real part of the matrix and one containing the
C  imaginary part.
C       When the logic variable LND equals FALSE the subroutine uses the
C  Crout algorithm to decompose [A] into [D].  [D] is a square matrix of
C  of the same size as [A], and replaces [A] so no additional memory is
C  needed.
C       When LND equals TRUE [D] is used to solve for [B].  [B] replaces
C  [C], again to save storage space.
C       A good reference for this method of solving matrix equations is
C  found in "Numerical Recipes : The Art of Scientific Computing", W.H.
C  Press, B.P. Flannery, S.A. Teukolsky, and  W.T. Vetterling, Cambridge
C  University Press, New York, 1986.
C       In terms of this program, two large matrices, one containing the
C  real parts and one containing the imaginary parts, of dimension
C  N x N+1 contain the square mutual impedance matrix for test dipoles
C  and the voltage excitation column matrix.  The values of this matrix
C  are replaced by the decomposed mutual impedance matrix and the
C  complex current column matrix.
C
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  IF
C                LND equals FALSE
C
C                    THEN
C
C                        Decompose square matrix using Crouts algorithm.
C
C                    ELSE
C
C                        Solve for column matrix using decomposed matrix.
C
C           ENDIF
C
C       2.  RETURN
C
C  INPUTS   EXPLICIT:
```

```
C       CI(I,J)    - imaginary values of mutual impedance matrix for test
C                    dipoles
C       CR(I,J)    - real values of mutual impedance matrix for test
C                    dipoles
C       CI(I,NVL) - imaginary values of excitation voltage column matrix
C       CR(I,NVL) - real values of excitation voltage column matrix
C       LND        - logical variable
C       N          - total # of test dipole modes
C       NVL        - column number for excitation voltage column matrix
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       CI(I,J)    - imaginary values of decomposed mutual impedance
C                    matrix for test dipoles
C       CR(I,J)    - real values of decomposed mutual impedance matrix
C                    for test dipoles
C       CI(I,NVL) - imaginary values of complex current column matrix
C       CR(I,NVL) - real values of complex current column matrix
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       See INPUTS and OUTPUTS.
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C    S. L. BERG, 12/05/88 - Added documentation.
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION CR(602,602),CI(602,602)
C
C Decompose square matrix using Crout algorithm.
C
      IF(LND) GO TO 630
      DO 118 L=1,N
      LLL=L-1
      DO 118 I=L,N
      II=I+1
      IF(LLL) 105,106,105
 105  DO 117 K=1,LLL
      CP1=CR(I,K)*CR(K,L)-CI(I,K)*CI(K,L)
      CR(I,L)=CR(I,L)-CP1
      CP1=CI(I,K)*CR(K,L)+CR(I,K)*CI(K,L)
      CI(I,L)=CI(I,L)-CP1
      CP1=CR(L,K)*CR(K,II)-CI(L,K)*CI(K,II)
      CR(L,II)=CR(L,II)-CP1
      CP1=CI(L,K)*CR(K,II)+CR(L,K)*CI(K,II)
 117  CI(L,II)=CI(L,II)-CP1
 106  CRP=CR(L,II)
      CLL=CR(L,L)*CR(L,L)+CI(L,L)*CI(L,L)
      CR(L,II)=CRP*CR(L,L)/CLL+CI(L,II)*CI(L,L)/CLL
 118  CI(L,II)=CI(L,II)*CR(L,L)/CLL-CRP*CI(L,L)/CLL
      DO 621 I=1,N
      CR(I,NVL)=0.
 621  CI(I,NVL)=0.
      LND=.TRUE.
```

```
      RETURN
 630  CONTINUE
C
C Solve for column matrix.
C
      DO 120 L=1,N
      LLL=L-1
      IF(LLL) 205,206,205
 205  DO 119 K=1,LLL
      CR(L,NVL)=CR(L,NVL)-CR(L,K)*CR(K,NVL)+CI(L,K)*CI(K,NVL)
 119  CI(L,NVL)=CI(L,NVL)-CI(L,K)*CR(K,NVL)-CR(L,K)*CI(K,NVL)
 206  CRP=CR(L,NVL)
      CLL=CR(L,L)*CR(L,L)+CI(L,L)*CI(L,L)
      CR(L,NVL)=(CRP*CR(L,L)+CI(L,NVL)*CI(L,L))/CLL
 120  CI(L,NVL)=(CI(L,NVL)*CR(L,L)-CRP*CI(L,L))/CLL
      DO 122 L=2,N
      I=NVL-L
      II=I+1
      DO 122 K=II,N
      CR(I,NVL)=CR(I,NVL)-CR(I,K)*CR(K,NVL)+CI(I,K)*CI(K,NVL)
 122  CI(I,NVL)=CI(I,NVL)-CI(I,K)*CR(K,NVL)-CR(I,K)*CI(K,NVL)
      RETURN
      END
C
C**********************************************************************
C+
      SUBROUTINE GSCRD(NW,NSW,WRL,SPC,ZHGT,INS,NM,NP,IA,IB,X,Y,Z)
C
C *********************************************************************
C *                                                                   *
C *    SUBROUTINE GSCRD                                               *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         08-FEBRUARY-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine determines the endpoints associated with
C  segments and their coordinates for a NAVSPASUR transmitter ground
C  screen.
C       The position of the segment endpoints with respect to the ground
C  screen are determined by the parameters of the ground screen as input
C  from the main program.  The east-west (E-W) segment length is fixed
C  by the number of segments per E-W wire and length of the E-W wires.
C  The north-south (N-S) segment length is determined by the spacing of
C  the E-W wires.  No provisions for more than one N-S segment between
C  the E-W wires has been made, due to the short distance between E-W
```

```
C  wires required for practical ground screens.
C       The endpoints are numbered starting with the endpoint with the
C  most negative x value on the first wire in the positive y direction
C  and progressing along the wire in the positive x direction.  The
C  endpoints on the next wire in the y direction follow in the same
C  manner, until all the endpoints with positive y values are numbered.
C  The endpoints with negative y values are then numbered, starting with
C  the most negative x value on the first wire in the negative y
C  direction and continuing along the wire.  The rest are numbered in
C  the same manner for each succesive wire in the negative y direction.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. DO
C               for each E-W segment
C
C                   Assign numbers to endpoints.
C
C          ENDDO
C
C       2. DO
C               for each endpoint.
C
C                   Assign coordinates.
C
C          ENDDO
C
C       3. DO
C
C               for each N-S wire
C
C                   Assign numbers to endpoints.
C
C          ENDDO
C
C       4. RETURN
C
C  INPUTS   EXPLICIT:
C       NW    - # of E-W ground wires
C       NSW   - # of segments/E-W wire
C       WRL   - E-W wire length
C       SPC   - spacing between E-W wires
C       ZHGT  - height of element vertex above ground screen
C       INS   - N-S wire indicator
C               (0 : no N-S wires; 1 : 13 N-S wires)
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       NM    - number of segments
C       NP    - number of points (segment endpoints)
C       IA(J) - 1st endpoint of segment J
C       IB(J) - 2nd endpoint of segment J
C       X(I)      - x coordinate of endpoint I (meters)
C       Y(I)      - y coordinate of endpoint I (meters)
C       Z(I)      - z coordinate of endpoint I (meters)
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
```

```
C       I     - endpoint number
C       J     - segment number
C       SGL   - segment length
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION IA(1),IB(1),X(1),Y(1),Z(1)
      NW2=NW/2
      RNW1=NW2
      RNW=NW
      RNW2=RNW/2.
      WRL2=WRL/2.
      NPW=NSW+1
      NM=NW*NSW
      NP=NW*NPW
      SGL=WRL/DFLOTJ(NSW)
      M=-1
C
C Set segment number to zero.
C
      J=0
C
C Match E-W segment numbers with endpoint numbers.  Each E-W wire has
C NSW segments and NPW endpoints.
C
      DO 100 I1=1,NW
       K=M+2
       DO 100 I2=1,NSW
        J=J+1
        M=K+I2-1
        IA(J)=M
        IB(J)=M+1
 100  CONTINUE
C
C Determine if the number of wires is odd or even for proper
C y coordinates of endpoints.  If NS is odd, the center wire is placed
C directly under element.  Otherwise the wires straddle the elements.
C
      YO=0.
      I=0
      IH1=(NW+1)/2
      IF(RNW2.EQ.RNW1) IH1=NW/2
      IF(RNW2.EQ.RNW1) YO=SPC/2.
      IH2=IH1*NPW
C
C Define segment endpoints by their coordinates.
C
      DO 200 I1=1,NW
       DO 200 I2=1,NPW
        I=I+1
        Z(I)=-ZHGT
        Y(I)=YO+DFLOTJ(I1-1)*SPC
```

```
          IF(I.GT.IH2) Y(I)=YO-DFLOTJ(I1-IH1)*SPC
          X(I)=-WRL2+(SGL*DFLOTJ(I2-1))
  200   CONTINUE
C
C Return to main program if no N-S wires exist.
C
        IF (INS.EQ.0) GO TO 900
C
C Define constants.
C
        NMA=NM
        NMB=NM+NPW
        K=0
C
C The N-S wires are assumed to intersect the E-W wires at all endpoint
C positions.
C
        DO 300 L=1,NPW
         J=J+1
         K=K+1
         IA(J)=K
         IB(J)=NP/2+K
  300   CONTINUE
        K=0
        NMC=NMB+(NW2-1)*NPW
        DO 400 L1=1,NW2-1
         DO 400 L2=1,NPW
          J=J+1
          K=K+1
          IA(J)=NPW+K
          IB(J)=K
  400   CONTINUE
        K=0
        DO 500 L1=1,NW2-1
         DO 500 L2=1,NPW
          J=J+1
          K=K+1
          IA(J)=NP/2+K
          IB(J)=NP/2+NPW+K
  500   CONTINUE
C
C Determine number of segments.
C
        NM=J
C
C Return to main program.
C
  900   RETURN
        END
C
C**********************************************************************
C+
        SUBROUTINE SORT4(IA,IB,I1,I2,I3,JA,JB,MD,ND,NM,NP,N,N1,N2,MAX,MIN,
       2ICJ,INS,INM,NW,NSW)
C
C *********************************************************************
C *                                                                   *
C *     SUBROUTINE SORT4                                              *
C *                                                                   *
C *********************************************************************
```

```
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         08-FEBRUARY-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine creates a set of test dipole modes for a
C  rectangular wire-grid screen, taking advantage of its 4-fold
C  symmetry.  A cartesian coordinate system is used, with the positive
C  x, y, and z axes corresponding to directions west, south, and up,
C  respectively.
C       The number of segments chosen per east-west wire can be odd or
C  even, if no north-south wires are used.  However, an odd number
C  allows for a totally four-fold symmetric set of test dipoles.  An
C  even number requires some test dipoles with only two-fold symmetry,
C  thereby reducing the efficency of the program.
C       If north-south wires are included, the subroutine is set for
C  only an even number of east-west segmnents.  The test dipoles are
C  fully four-fold symmetric.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  IF
C                # of segments per wire is even
C
C                   THEN
C
C                        IF
C                           north-south wires exist
C
C                                THEN
C                                    for each quadrant.
C
C                                        Determine test dipoles.
C
C                                        Determine corresponding
C                                        segments.
C
C                                ELSE
C                                    for each quadrant.
C
C                                        Determine 4-fold symmetric test
C                                        dipoles.
C
C                                        Determine corresponding
C                                        segments.
C
C                                    for each half (south and north).
C
```

```
C                                   Determine 2-fold symmetric test
C                                   dipoles.
C
C                                   Determine corresponding
C                                   segments.
C                    ENDIF
C
C                ELSE
C                    for each quadrant
C
C                      Determine test dipoles.
C
C                      Determine corresponding segments.
C
C         ENDIF
C
C      2. RETURN
C
C
C  INPUTS   EXPLICIT:
C      IA(J)    - 1st endpoint of segment J
C      IB(J)    - 2nd endpoint of segment J
C      ICJ      - maximum allowed number of test dipole modes
C      INM      - maximum allowed number of segments
C      INS      - N-S wire indicator
C                 (0 : no N-S wires; 1 : 13 N-S wires)
C      NM       - number of segments
C      NP       - number of points (segment endpoints)
C      NSW      - # of segments/E-W wire
C      NW       - # of E-W ground wires
C
C           IMPLICIT:
C
C  OUTPUTS  EXPLICIT:
C      I1(I)    - 1st endpoint of test dipole I
C      I2(I)    - terminal endpoint of test dipole I
C      I3(I)    - 2nd endpoint of test dipole I
C      JA(I)    - 1st segment of test dipole I
C      JB(I)    - 2nd segment of test dipole I
C      MD(J,K)  - list of test dipoles sharing segment J
C      MAX      - maximum value of ND(J)
C      MIN      - minimum value of ND(J)
C      N        - total # of test dipole modes
C      N1       - # of test dipole modes in one quadrant of the ground
C                 screen, which have symmetric counter parts in the
C                 other three quadrants.
C      N2       - # of test dipole modes in one half of the ground
C                 screen, which have symmetric counter parts in the
C                 other half.
C      ND(J)    - total number of test dipoles sharing segments J
C
C           IMPLICIT:
C
C  OTHER MAJOR VARIABLES:
C
C      IOE      - odd-even indicator (1 : odd; 2 : even)
C      ND4      - # of test dipoles divided by four
C      NPW      - # of points per wire
C
C  SYSTEM STATE CHANGES:  NONE
```

```
C
C-
C
C  MODIFIED:
C
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      DIMENSION I1(1),I2(1),I3(1),JA(1),JB(1)
      DIMENSION IA(1),IB(1),ND(1),MD(INM,4)
C
C Define variables.
C
      NW2=NW/2
      NSW2=NSW/2
      ND4=(NSW-1)/2
C
C Determine if NSW is odd or even.
C
      IOE=1
      IF(2*NSW2.EQ.NSW) THEN
        IOE=2
        ND4=(NSW-2)/2
      ENDIF
      NPW=NSW+1
      I=0
C
C  This section creates the test dipoles if no N-S wires are present.
C
      IF(INS.NE.0) GO TO 100
C
C  Determine the test dipole points in first quadrant.
C  All the test dipoles are straight.
C
      DO 10 K=1,NW2
       DO 10 L=1,ND4
        I=I+1
        K1=(K-1)*NPW
        I1(I)=K1+L
        I2(I)=K1+L+1
        I3(I)=K1+L+2
 10   CONTINUE
C
C  Determine N1.
C
      N1=I
C
C  Determine the test dipole points in second quadrant.
C  All the test dipoles are straight.
C
      DO 12 K=1,NW2
       DO 12 L=1,ND4
        I=I+1
        K1=(K-1)*NPW+NPW
        I1(I)=K1-L-1
        I2(I)=K1-L
        I3(I)=K1-L+1
 12   CONTINUE
C
```

```
C  Determine the test dipole points in third quadrant.
C  All the test dipoles are straight.
C
      DO 14 K=1,NW2
       DO 14 L=1,ND4
        I=I+1
        K1=NPW*NW2+(K-1)*NPW
        I1(I)=K1+L
        I2(I)=K1+L+1
        I3(I)=K1+L+2
 14   CONTINUE
C
C  Determine the test dipole points in fourth quadrant.
C  All the test dipoles are straight.
C
      DO 16 K=1,NW2
       DO 16 L=1,ND4
        I=I+1
        K1=NPW*NW2+(K-1)*NPW+NPW
        I1(I)=K1-L-1
        I2(I)=K1-L
        I3(I)=K1-L+1
 16   CONTINUE
C
C  Determine N and N2.
C
      N=I
      N2=N
C
C  If the number of segments per wires is even, a test dipole straddles
C  the east and west halves and must be dealt with differently since it
C  has only two-fold symmetry.
C
      IF(IOE.EQ.1) GO TO 20
      DO 18 K=1,NW
       I=I+1
       K1=(NPW-1)/2+(K-1)*NPW
       I1(I)=K1
       I2(I)=K1+1
       I3(I)=K1+2
 18   CONTINUE
C
C  Determine N and N2.
C
      N=I
      N2=N-NW2
C
C  Determine the segments corresponding to the test dipoles.
C
 20   I=0
C
C  Determine the test dipole segments in first quadrant.
C
      DO 30 K=1,NW2
       DO 30 L=1,ND4
        I=I+1
        K1=(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K1+L+1
 30   CONTINUE
```

```
C
C  Determine the test dipole segments in second quadrant.
C
      DO 32 K=1,NW2
       DO 32 L=1,ND4
        I=I+1
        K1=(K-1)*NSW+NSW
        JA(I)=K1-L
        JB(I)=K1-L+1
 32   CONTINUE
C
C  Determine the test dipole segments in third quadrant.
C
      DO 34 K=1,NW2
       DO 34 L=1,ND4
        I=I+1
        K1=NSW*NW2+(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K1+L+1
 34   CONTINUE
C
C  Determine the test dipole segments in fourth quadrant.
C
      DO 36 K=1,NW2
       DO 36 L=1,ND4
        I=I+1
        K1=NSW*NW2+(K-1)*NSW+NSW
        JA(I)=K1-L
        JB(I)=K1-L+1
 36   CONTINUE
C
C  If the number of segments per wires is even, a test dipole straddles
C  the east and west halves and must be dealt with differently since it
C  has only two-fold symmetry.
C
      IF (IEO.EQ.1) GO TO 340
      DO 38 K=1,NW
       I=I+1
       K1=NSW2+(K-1)*NSW
       JA(I)=K1
       JB(I)=K1+1
 38   CONTINUE
      GO TO 340
C
C  This section creates the test dipoles if N-S wires are present.
C
C   Determine the test dipole points in first quadrant.
C
C    These test dipoles are straight oriented east-west.
C
 100  DO 101 K=1,NW2
       DO 101 L=1,ND4
        I=I+1
        K1=(K-1)*NPW
        I1(I)=K1+L
        I2(I)=K1+L+1
        I3(I)=K1+L+2
 101  CONTINUE
C
C    These test dipoles have right angles positioned in southeast
```

```
C    corner of the grid rectangle.
C
      DO 102 K=1,NW2
       DO 102 L=1,NSW2
        I=I+1
        K1=(K-2)*NPW
        IF(K.EQ.1) K1=NPW*NW2
        K2=(K-1)*NPW
        I1(I)=K1+L
        I2(I)=K2+L
        I3(I)=K2+L+1
 102  CONTINUE
C
C    These test dipoles are straight oriented north-south.
C
      DO 103 K=1,NW2-1
       DO 103 L=1,NSW2
        I=I+1
        K1=(K-2)*NPW
        IF(K.EQ.1) K1=NPW*NW2
        K2=(K-1)*NPW
        K3=K*NPW
        I1(I)=K1+L
        I2(I)=K2+L
        I3(I)=K3+L
 103  CONTINUE
C
C    These test dipoles have right angles positioned in southwest
C    corner of the grid rectangle.
C
      DO 104 K=1,NW2
       I=I+1
       K1=(K-1)*NPW+ND4+2
       K3=(K-2)*NPW+ND4+2
       IF(K.EQ.1) K3=NW2*NPW+ND4+2
       I1(I)=K1-1
       I2(I)=K1
       I3(I)=K3
 104  CONTINUE
C
C    These test dipoles have right angles positioned in northwest
C    corner of the grid rectangle.
C
      DO 105 K=1,NW2-1
       I=I+1
       K1=(K-1)*NPW+ND4+2
       K3=K*NPW+ND4+2
       I1(I)=K1-1
       I2(I)=K1
       I3(I)=K3
 105  CONTINUE
C
C  Determine N1.
C
      N1=I
C
C   Determine the test dipole points in second quadrant.
C
C    These test dipoles are straight oriented east-west.
C
```

```
      DO 111 K=1,NW2
       DO 111 L=1,ND4
        I=I+1
        K1=K*NPW
        I1(I)=K1-L+1
        I2(I)=K1-L
        I3(I)=K1-L-1
 111  CONTINUE
C
C    These test dipoles have right angles positioned in southwest
C    corner of the grid rectangle.
C
      DO 112 K=1,NW2
       DO 112 L=1,NSW2
        I=I+1
        K1=(K-1)*NPW
        IF(K.EQ.1) K1=NPW*NW2+NPW
        K2=K*NPW
        I1(I)=K1-L+1
        I2(I)=K2-L+1
        I3(I)=K2-L
 112  CONTINUE
C
C    These test dipoles are straight oriented north-south.
C
      DO 113 K=1,NW2-1
       DO 113 L=1,NSW2
        I=I+1
        K1=(K-1)*NPW
        IF(K.EQ.1) K1=NPW*NW2+NPW
        K2=K*NPW
        K3=(K+1)*NPW
        I1(I)=K1-L+1
        I2(I)=K2-L+1
        I3(I)=K3-L+1
 113  CONTINUE
C
C    These test dipoles have right angles positioned in southeast
C    corner of the grid rectangle.
C
      DO 114 K=1,NW2
       I=I+1
       K1=K*NPW-ND4
       K3=(K-1)*NPW-ND4-1
       IF(K.EQ.1) K3=NW2*NPW+NPW-ND4-1
       I1(I)=K1
       I2(I)=K1-1
       I3(I)=K3
 114  CONTINUE
C
C    These test dipoles have right angles positioned in northeast
C    corner of the grid rectangle.
C
      DO 115 K=1,NW2-1
       I=I+1
       K1=K*NPW-ND4
       K3=(K+1)*NPW-ND4-1
       I1(I)=K1
       I2(I)=K1-1
       I3(I)=K3
```

```
  115 CONTINUE
C
C   Determine the test dipole points in third quadrant.
C
C    These test dipoles are straight oriented east-west.
C
      DO 121 K=1,NW2
       DO 121 L=1,ND4
        I=I+1
        K1=(K-1)*NPW+NW2*NPW
        I1(I)=K1+L
        I2(I)=K1+L+1
        I3(I)=K1+L+2
  121 CONTINUE
C
C    These test dipoles have right angles positioned in northeast
C    corner of the grid rectangle.
C
      DO 122 K=1,NW2
       DO 122 L=1,NSW2
        I=I+1
        K1=(K-2)*NPW+NW2*NPW
        IF(K.EQ.1) K1=0
        K2=(K-1)*NPW+NW2*NPW
        I1(I)=K1+L
        I2(I)=K2+L
        I3(I)=K2+L+1
  122 CONTINUE
C
C    These test dipoles are straight oriented north-south.
C
      DO 123 K=1,NW2-1
       DO 123 L=1,NSW2
        I=I+1
        K1=(K-2)*NPW+NW2*NPW
        IF(K.EQ.1) K1=0
        K2=(K-1)*NPW+NW2*NPW
        K3=K*NPW+NW2*NPW
        I1(I)=K1+L
        I2(I)=K2+L
        I3(I)=K3+L
  123 CONTINUE
C
C    These test dipoles have right angles positioned in northwest
C    corner of the grid rectangle.
C
      DO 124 K=1,NW2
       I=I+1
       K1=(K-1)*NPW+NW2*NPW+ND4+2
       K3=(K-2)*NPW+NW2*NPW+ND4+2
       IF(K.EQ.1) K3=ND4+2
       I1(I)=K1-1
       I2(I)=K1
       I3(I)=K3
  124 CONTINUE
C
C    These test dipoles have right angles positioned in southwest
C    corner of the grid rectangle.
C
      DO 125 K=1,NW2-1
```

```
      I=I+1
      K1=(K-1)*NPW+NW2*NPW+ND4+2
      K3=K*NPW+NW2*NPW+ND4+2
      I1(I)=K1-1
      I2(I)=K1
      I3(I)=K3
 125  CONTINUE
C
C   Determine the test dipole points in fourth quadrant.
C
C    These test dipoles are straight oriented east-west.
C
      DO 131 K=1,NW2
       DO 131 L=1,ND4
        I=I+1
        K1=K*NPW+NW2*NPW
        I1(I)=K1-L+1
        I2(I)=K1-L
        I3(I)=K1-L-1
 131  CONTINUE
C
C    These test dipoles have right angles positioned in northwest
C    corner of the grid rectangle.
C
      DO 132 K=1,NW2
       DO 132 L=1,NSW2
        I=I+1
        K1=(K-1)*NPW+NW2*NPW
        IF(K.EQ.1) K1=NPW
        K2=K*NPW+NW2*NPW
        I1(I)=K1-L+1
        I2(I)=K2-L+1
        I3(I)=K2-L
 132  CONTINUE
C
C    These test dipoles are straight oriented north-south.
C
      DO 133 K=1,NW2-1
       DO 133 L=1,NSW2
        I=I+1
        K1=(K-1)*NPW+NW2*NPW
        IF(K.EQ.1) K1=NPW
        K2=K*NPW+NW2*NPW
        K3=(K+1)*NPW+NW2*NPW
        I1(I)=K1-L+1
        I2(I)=K2-L+1
        I3(I)=K3-L+1
 133  CONTINUE
C
C    These test dipoles have right angles positioned in northeast
C    corner of the grid rectangle.
C
      DO 134 K=1,NW2
       I=I+1
       K1=K*NPW+NW2*NPW-ND4
       K3=(K-1)*NPW+NW2*NPW-ND4-1
       IF(K.EQ.1) K3=NPW-ND4-1
       I1(I)=K1
       I2(I)=K1-1
       I3(I)=K3
```

```
 134  CONTINUE
C
C    These test dipoles have right angles positioned in southeast
C    corner of the grid rectangle.
C
      DO 135 K=1,NW2-1
       I=I+1
       K1=K*NPW+NW2*NPW-ND4
       K3=(K+1)*NPW+NW2*NPW-ND4-1
       I1(I)=K1
       I2(I)=K1-1
       I3(I)=K3
 135  CONTINUE
C
C  Determine N2 and N.
C
      N2=I
      N=I
C
C  Determine the segments corresponding to the test dipoles.
C
      I=0
C
C  Determine the test dipole segments in first quadrant.
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented east-west.
C
 200  DO 201 K=1,NW2
       DO 201 L=1,ND4
        I=I+1
        K1=(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K1+L+1
 201  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the southeast corner of the grid rectangle.
C
      DO 202 K=1,NW2
       DO 202 L=1,NSW2
        I=I+1
        K1=NW*NSW+(K-1)*NPW
        K2=(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K2+L
 202  CONTINUE
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented north-south.
C
      DO 203 K=1,NW2-1
       DO 203 L=1,NSW2
        I=I+1
        K1=NW*NSW+(K-1)*NPW
        K2=NW*NSW+K*NPW
        JA(I)=K1+L
        JB(I)=K2+L
 203  CONTINUE
C
```

```
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the southwest corner of the grid rectangle.
C
      DO 204 K=1,NW2
       I=I+1
       K1=(K-1)*NSW+NSW2
       K2=NW*NSW+(K-1)*NPW+NSW2+1
       JA(I)=K1
       JB(I)=K2
 204  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northwest corner of the grid rectangle.
C
      DO 205 K=1,NW2-1
       I=I+1
       K1=(K-1)*NSW+NSW2
       K2=NW*NSW+K*NPW+NSW2+1
       JA(I)=K1
       JB(I)=K2
 205  CONTINUE
C
C  Determine the test dipole segments in second quadrant.
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented east-west.
C
      DO 211 K=1,NW2
       DO 211 L=1,ND4
        I=I+1
        K1=(K-1)*NSW+NSW
        JA(I)=K1-L+1
        JB(I)=K1-L
 211  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the southwest corner of the grid rectangle.
C
      DO 212 K=1,NW2
       DO 212 L=1,NSW2
        I=I+1
        K1=NW*NSW+K*NPW
        K2=(K-1)*NSW+NSW
        JA(I)=K1-L+1
        JB(I)=K2-L+1
 212  CONTINUE
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented north-south.
C
      DO 213 K=1,NW2-1
       DO 213 L=1,NSW2
        I=I+1
        K1=NW*NSW+K*NPW
        K2=NW*NSW+(K+1)*NPW
        JA(I)=K1-L+1
        JB(I)=K2-L+1
 213  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
```

```
C  positioned in the southeast corner of the grid rectangle.
C
      DO 214 K=1,NW2
       I=I+1
       K1=K*NSW-NSW2+1
       K2=NW*NSW+K*NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 214  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northeast corner of the grid rectangle.
C
      DO 215 K=1,NW2-1
       I=I+1
       K1=K*NSW-NSW2+1
       K2=NW*NSW+(K+1)*NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 215  CONTINUE
C
C  Determine the test dipole segments in third quadrant.
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented east-west.
C
      DO 221 K=1,NW2
       DO 221 L=1,ND4
        I=I+1
        K1=NSW*NW2+(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K1+L+1
 221  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northeast corner of the grid rectangle.
C
      DO 222 K=1,NW2
       DO 222 L=1,NSW2
        I=I+1
        K1=NW*NSW+NW2*NPW+(K-2)*NPW
        IF(K.EQ.1) K1=NW*NSW
        K2=NSW*NW2+(K-1)*NSW
        JA(I)=K1+L
        JB(I)=K2+L
 222  CONTINUE
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented north-south.
C
      DO 223 K=1,NW2-1
       DO 223 L=1,NSW2
        I=I+1
        K1=NW*NSW+NW2*NPW+(K-2)*NPW
        IF(K.EQ.1) K1=NW*NSW
        K2=NW*NSW+NW2*NPW+(K-1)*NPW
        JA(I)=K1+L
        JB(I)=K2+L
 223  CONTINUE
C
```

```
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northwest corner of the grid rectangle.
C
      DO 224 K=1,NW2
       I=I+1
       K1=NSW*NW2+(K-1)*NSW+NSW2
       K2=NW*NSW+NW2*NPW+(K-1)*NPW-NSW2
       IF(K.EQ.1) K2=NW*NSW+NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 224  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the southwest corner of the grid rectangle.
C
      DO 225 K=1,NW2-1
       I=I+1
       K1=NSW*NW2+(K-1)*NSW+NSW2
       K2=NW*NSW+NW2*NPW+K*NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 225  CONTINUE
C
C  Determine the test dipole segments in fourth quadrant.
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented east-west.
C
      DO 231 K=1,NW2
       DO 231 L=1,ND4
        I=I+1
        K1=NSW*NW2+K*NSW
        JA(I)=K1-L+1
        JB(I)=K1-L
 231  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northwest corner of the grid rectangle.
C
      DO 232 K=1,NW2
       DO 232 L=1,NSW2
        I=I+1
        K1=NW*NSW+NW2*NPW+(K-1)*NPW
        IF(K.EQ.1) K1=NW*NSW+NPW
        K2=NSW*NW2+K*NSW
        JA(I)=K1-L+1
        JB(I)=K2-L+1
 232  CONTINUE
C
C  Determine the segments corresponding to the straight test dipoles
C  oriented north-south.
C
      DO 233 K=1,NW2-1
       DO 233 L=1,NSW2
        I=I+1
        K1=NW*NSW+NW2*NPW+(K-1)*NPW
        IF(K.EQ.1) K1=NW*NSW+NPW
        K2=NW*NSW+NW2*NPW+K*NPW
        JA(I)=K1-L+1
        JB(I)=K2-L+1
```

```
 233  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the northeast corner of the grid rectangle.
C
      DO 234 K=1,NW2
       I=I+1
       K1=NSW*NW2+K*NSW-NSW2+1
       K2=NW*NSW+NW2*NPW+(K-1)*NPW-NSW2
       IF(K.EQ.1) K2=NW*NSW+NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 234  CONTINUE
C
C  Determine the segments corresponding to the right angle test dipoles
C  positioned in the southeast corner of the grid rectangle.
C
      DO 235 K=1,NW2-1
       I=I+1
       K1=NSW*NW2+K*NSW-NSW2+1
       K2=NW*NSW+NW2*NPW+K*NPW-NSW2
       JA(I)=K1
       JB(I)=K2
 235  CONTINUE
C
C  Set initial values of ND(J) and MD(J,K) equal to zero.
C
 340  DO 350 J=1,NM
       ND(J)=0
       DO 350 K=1,4
        MD(J,K)=0
 350  CONTINUE
      III=N
      IF(N.GT.ICJ)III=ICJ
C
C  Determine ND(J) and MD(J,K).
C
      DO 360 I=1,III
       J=JA(I)
       DO 358 L=1,2
        ND(J)=ND(J)+1
        K=1
        M=0
 352    MJK=MD(J,K)
        IF(MJK.NE.0)GO TO 354
        M=1
        MD(J,K)=I
 354    K=K+1
        IF(K.GT.4)GO TO 358
        IF(M.EQ.0)GO TO 352
 358    J=JB(I)
 360  CONTINUE
C
C  Determine MAX and MIN.
C
      MIN=100
      MAX=0
      DO 366 J=1,NM
       NDJ=ND(J)
       IF(NDJ.GT.MAX)MAX=NDJ
```

```
      IF(NDJ.LT.MIN)MIN=NDJ
 366  CONTINUE
C
C  Return to main program.
C
      RETURN
      END
```

```
$ !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ !                                                                     !
$ !                      GEN_ARRAY.COM                                  !
$ !                                                                     !
$ !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
$ SET DEF SPACE:[BERG.ARRAY]
$ !
$ ! Inquire name of main program.
$ !
$ IF P1.EQS."" THEN INQUIRE P1 "PROGRAM"
$ !
$ ! Inquire file names of ELPOS  - element positions in array
$ !                      CURRNT - terminal current values of each
$ !                               element
$ !                      ELFFLD - element electric field values
$ !      The names of the subroutines are given above, however each
$ !      different subroutine is given a unique file name, usually in
$ !      the form ELPOS_****, CURRNT_****, or ELFFLD_**** to specify
$ !      the related subroutine's name.
$ !
$ IF P2.EQS."" THEN INQUIRE P2 "ELPOS SUBROUTINE FILE NAME"
$ IF P3.EQS."" THEN INQUIRE P3 "CURRNT SUBROUTINE FILE NAME"
$ IF P4.EQS."" THEN INQUIRE P4 "ELFFLD SUBROUTINE FILE NAME"
$ !
$ ASSIGN 'P1'.OUT FOR007
$ !
$ ! Each subroutine must have a corresponding command file which assign
$ ! the input and output files the proper FORTRAN unit number.
$ !
$ @'P1'.COM
$ @'P2'.COM
$ @'P3'.COM
$ @'P4'.COM
$ SET VERIFY
$ !
$ ! Compile *.FOR files.
$ !
$ FORTRAN/CHECK 'P1'
$ FORTRAN/CHECK ARRAY_SBR
$ FORTRAN/CHECK 'P2'
$ FORTRAN/CHECK 'P3'
$ FORTRAN/CHECK 'P4'
$ !
$ ! Link *.OBJ files.
$ !
$ LINK 'P1',ARRAY_SBR,'P2','P3','P4'
$ !
$ ! Delete *.OBJ files
$ !
$ DELETE *.OBJ;*
$ !
$ ! Run program.
$ !
$ RUN 'P1'
$ SET NOVERIFY
```

```
C+
      PROGRAM NEARFLD
C
C ***********************************************************************
C *                                                                     *
C *     PROGRAM NEARFLD                                                 *
C *                                                                     *
C ***********************************************************************
C
C  AUTHOR:      DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:        01-AUGUST-1988
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.ARRAY]NEARFLD.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  ARRAY_SBR  - calculates the electric field at a
C                                    given point in space for a selected
C                                    array.
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN NEARFLD
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK NEARFLD,ARRAY_SBR,'ELPOS',
C                                      'CURRNT','ELFFLD'
C
C       NOTE: This program has been designed as a part of a set of
C             programs and subroutines to be used with the command file
C             MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  PARENT PROGRAM:      MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  PROGRAM DESCRIPTION:
C       This program calculates the selected electric field and power
C  density values of an array of radiating elements for a selected range
C  and plane.
C       The coordinate system chosen for the inputs places the x, y, and
C  z axes along the directions vertical, west, and south, respectively.
C  This is not the same coordinate system as is used by the subroutine
C  ARRAY_SBR, but takes advantage of the symmetry of the NAVSPASUR
C  antennas, which are aligned north-south.
C       Far field (infinite range) calculations can be performed by
C  setting the range equal to zero, since zero range is never needed for
C  practical calculations.  The far electric field has the range factor
C  exp(ikr)/r removed, and the power has the 1/r*r term removed.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C       1.  READ
C               - antenna range
C               - # of theta values
C               - initial theta value
C               - theta interval
C               - # of phi values
C               - initial phi value
C               - phi interval
C
C       2.  DO
C               for each value of theta.
C
C               DO
C                   for each value of phi.
```

```
C
C                        Perform coordinate transformation.
C
C                        CALL
C                            cartesian components of electric field.
C
C                        Determine electric field magnitude.
C
C                        Determine power density.
C
C                        Determine electric field phase.
C
C                        WRITE
C                             - phi value
C                             - theta value
C                             - electric field magnitude
C                             - electric field phase
C                             - power density
C
C                ENDDO
C
C            ENDDO
C
C  INPUTS   EXPLICIT:
C       FOR022 - NEARFLD.INP
C               NOPHVL  - # of phi values
C               NOTHVL  - # of theta values
C               PHIO    - initial phi value (degrees)
C               PHIINT  - phi step interval (degrees)
C               R       - antenna range (kilometers)
C               THETAO  - intial theta value (degrees)
C               THETINT - theta step interval (degrees)
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       FOR007 - NEARFLD.OUT
C               AEFMG   - array electric field magnitude (Volts/meter)
C               AEFPH   - array electric field phase (degrees)
C               DPH     - phi (degrees)
C               DTH     - theta (degrees)
C               POWER   - power density (Watts/square meter)
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       AEF     - complex array electric field
C       AEFX    - x component of complex array electric field
C       AEFY    - y component of complex array electric field
C       AEFZ    - z component of complex array electric field
C       CFDR    - conversion factor : degrees to radians
C       ETA     - intrinsic impedance of ambient medium
C       IFLAG   - flag (0 : first loop, 1 : subsequent loops)
C       IUNIT   - output file FORTRAN unit number
C       PHPR    - phi value in subroutine coordinate system
C       THPR    - theta value in subroutine coordinate system
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
```

```
C
C  MODIFIED:
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 AEF,AEFX,AEFY,AEFZ
C
C Define constants.
C
      PI=3.141592653589793
      CFDR=PI/180.
      ETA=376.730311
      PI2=PI/2.
C
C Set flag.
C
      IFLAG=0
C
C Set NEARFLD.OUT FORTRAN unit number.
C
      IUNIT=7
C
C Read input values.
C
      READ(22,*) R
      READ(22,*) NOTHVL
      READ(22,*) THETAO
      READ(22,*) THETINT
      READ(22,*) NOPHVL
      READ(22,*) PHIO
      READ(22,*) PHIINT
      CLOSE(22)
C
C Loop through all values of theta.
C
      DO I=1,NOTHVL
       DTH=THETAO+(I-1.)*THETINT
       TH=DTH*CFDR
C
C Calculate cosine and sine.
C
       CTH=DCOS(TH)
       STH=DSIN(TH)
C
C Adjust for precision errors.
C
       IF(DABS(CTH).LT.1.D-15) CTH=0.
       IF(DABS(STH).LT.1.D-15) STH=0.
C
C Loop through all values of phi.
C
       DO J=1,NOPHVL
        DPH=PHIO+(J-1.)*PHIINT
        PH=DPH*CFDR
C
C Determine cosine and sine.
C
        CPH=DCOS(PH)
        SPH=DSIN(PH)
```

```
C
C Adjust for precision errors.
C
      IF(DABS(CPH).LT.1.D-15) CPH=0.
      IF(DABS(SPH).LT.1.D-15) SPH=0.
C
C Perform coordinate transformations.
C
      THPR=DACOS(STH*CPH)
      IF(CTH.EQ.0. .AND. SPH.EQ.0.) THEN
        IF(CPH.LE.0.) THEN
          PHPR=PI
         ELSE
          PHPR=0.
        ENDIF
       ELSE
        PHPR=DATAN2(CTH,STH*SPH)
      ENDIF
C
C Convert transformed coordinates from radians to degrees.
C
      DTHETA=THPR/CFDR
      DPHI=PHPR/CFDR
C
C Determine cartesian components of the array electric field at the
C point (R,THPR,PHPR) in space.
C
      CALL ARRAY_SBR(R,THPR,PHPR,IFLAG,IUNIT,AEFX,AEFY,AEFZ)
C
C Write header lines.
C
      IF (IFLAG.EQ.0) THEN
        WRITE(7,1005)
        WRITE(7,1001)
        WRITE(7,1006)
      ENDIF
C
C Reset flag.
C
      IFLAG=1
C
C Determine electric field magnitude.
C
      AEF=CDSQRT((AEFX)*(AEFX)+(AEFY)*(AEFY)+(AEFZ)*(AEFZ))
      AEFMG=CDABS(AEF)
C
C Determine power density.
C
      POWER=AEFMG*AEFMG/2./ETA
C
C Determine real and imaginary parts of cartesion components of electric
C field.
C
      AEFXR=DREAL(AEFX)
      AEFYR=DREAL(AEFY)
      AEFZR=DREAL(AEFZ)
      AEFXI=DIMAG(AEFX)
      AEFYI=DIMAG(AEFY)
      AEFZI=DIMAG(AEFZ)
C
```

```
C Set up positive and negative regions of space to use to determine the
C phase of the electric field.
C
      IF(AEFXR.GT.0.) SIGNR=1.0
        IF(AEFXR.LT.0.) SIGNR=-1.0
        IF(AEFXR.EQ.0.) THEN
          IF(AEFZR.GT.0.) SIGNR=1.0
          IF(AEFZR.LT.0.) SIGNR=-1.0
          IF(AEFZR.GT.0.) THEN
            IF(AEFYR.GT.0.) SIGNR=1.0
            IF(AEFYR.LT.0.) SIGNR=-1.0
          ENDIF
        ENDIF
C
C Determine the phase of the electric field.
C
        AEF=SIGNR*AEF
        AEFR=DREAL(AEF)
        AEFI=DIMAG(AEF)
        IF(AEFI.NE.0. .OR. AEFR.NE.0) THEN
         AEFPH=DATAN2(AEFI,AEFR)/CFDR
        ENDIF
C
C Write electric field magnitude and phase and power density for this
C point in space (R,DTH,DPH).
C
        WRITE(7,1007)DPH,DTH,AEFMG,AEFPH,POWER
       ENDDO
      ENDDO
C
C Format statements.
C
 1001 FORMAT(2X,'PHI',7X,'THETA',5X,'E',22X,'POWER')
 1005 FORMAT('1')
 1006 FORMAT(3X,'(deg.)',4X,'(deg.)',3X,'magnitude',3X,'phase')
 1007 FORMAT(1X,F9.4,2X,F8.4,2X,E9.3,2X,F8.3,2X,E9.3)
C
C End of program.
C
      STOP
      END
C************************************************************************
```

```
C+
      SUBROUTINE ARRAY_SBR(R,THETA,PHI,IFLAG,IUNIT,AEFX,AEFY,AEFZ)
C
C *********************************************************************
C *                                                                   *
C *   SUBROUTINE ARRAY_SBR                                            *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:      DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:        27-MAY-1988
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.ARRAY]ARRAY_SBR.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  ELPOS  - calculates positions for each element
C                                terminal of the specified array.
C                       CURRNT - calculates the complex current terminal
C                                value for each element of the specified
C                                array.
C                       ELFFLD - calculates for a specified direction
C                                the element's range-independent far
C                                electric field for unit terminal
C                                current.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ARRAY_SBR
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       'CURRNT','ELFFLD'
C
C  PARENT PROGRAM:   'MAIN'.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the electric field at a specified
C  angle for a specified array, element pattern, terminal current
C  distribution, and range.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. IF
C               this is the first time the subroutine is called
C
C                   THEN
C
C                       WRITE
C                           - antenna range value
C
C                       CALL
C                           - element positions
C
C                       WRITE
C                           - array name
C
C                       CALL
C                           - complex current values
C
C                       CALL
C                           - element heading
C
```

```
C          ENDIF
C
C       2.  DO
C              - for each element in the array.
C
C                Determine the angles with respect to the terminal.
C
C                CALL
C                    - element's far electric field components, less
C                      range and current dependence.
C
C                Include range and current dependence.
C
C                Superimpose electric field contribution to previous
C                elements' electric field contribution.
C
C           ENDDO
C
C       3.  RETURN
C
C
C  INPUTS   EXPLICIT:
C       IFLAG    - flag : 0 => first time called
C                         1 => previously called
C       IUNIT    - output unit number
C       PHI      - standard spherical coordinates phi (radians)
C       R        - beam pattern range (kilometers); R=0 => far field
C                  (infinite range), range dependence suppressed
C       THETA    - angle theta (radians)
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       AEFX     - x component of electric field of array (Volts/meter)
C       AEFY     - y component of electric field of array (Volts/meter)
C       AEFZ     - z component of electric field of array (Volts/meter)
C
C       FOR007 - 'MAIN'.OUT
C               NAME    - name of array
C               R       - range (kilometers)
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       CFDR     - conversion factor: degrees to radians
C       CRNT(K)  - complex current at element K
C       EFLDX    - x component of far electric field from element with
C                  ground screen with range andcurrent dependence.
C       EFLDY    - y component of far electric field from element with
C                  ground screen with range andcurrent dependence.
C       EFLDZ    - z component of far electric field from element with
C                  ground screen with range andcurrent dependence.
C       EFX      - x component of far electric field from element with
C                  range and current dependence suppressed.
C       EFY      - y component of far electric field from element with
C                  range and current dependence suppressed.
C       EFZ      - z component of far electric field from element with
C                  range and current dependence suppressed.
C       ETA      - intrinsic impedance of ambient medium
C       GAM      - complex propagation constant of ambient medium
```

```
C       JFLAG   - flag : 0 => first time called
C                        1 => previously called
C       NEL     - # of elements in array
C       PHIPP   - angle phi of point in space, origin : array element
C                 terminal (axes parallel to primary coordinate system)
C       PPP     - rho coordinate of point in space wrt array element
C                 terminal, origin : array element terminal, axes : same
C                 direction as primary coordinate system
C       PTHDIF  - path difference for array far field calculation
C       RNGFTR  - range dependence of electric field
C       RPP     - r coordinate of point in space wrt array element
C                 terminal, origin : array element terminal, axes : same
C                 direction as primary coordinate system
C       THETPP  - angle theta of point in space, origin : array element
C                 terminal (axes parallel to primary coordinate system)
C       X       - x coordinate of point in space at which electric field
C                 is being calculated (primarycoordinate system)
C       XP      - x coordinate of array element terminal (primary
C                 coordinate system)
C       Y       - y coordinate of point in space at which electric field
C                 is being calculated (primarycoordinate system)
C       YP(K)   - y coordinate of array element terminal (primary
C                 coordinate system)
C       Z       - z coordinate of point in space at which electric field
C                 is being calculated (primarycoordinate system)
C       ZP      - z coordinate of array element terminal (primary
C                 coordinate system)
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 EFLDX,EFLDY,EFLDZ,EFX,EFY,EFZ,AEFX,AEFY,AEFZ
      COMPLEX*16 ETA,GAM,PTHDIF,RNGFTR
      COMPLEX*16 CRNT(2592)
      CHARACTER*30 NAME
      DIMENSION  YP(2592)
C
C Set values of constants.
C
      DATA PI/3.141592653589793/
      IF(IFLAG.EQ.0) THEN
       CFDR=PI/180.DO
       IUNIT=7
       JFLAG=0
C
C Write range value.
C
       IF(R.NE.0.) THEN
         WRITE(IUNIT,1001) R
        ELSE
         WRITE(IUNIT,1002)
       ENDIF
C
C Convert range to meters.
C
       R=1.D3*R
```

```
C
C Calculate element positions.
C
       CALL ELPOS(NAME,NEL,XP,YP,ZP)
C
C Write array name.
C
       WRITE(IUNIT,101) NAME
C
C Call a set of complex current values for each element terminal.
C
       CALL CURRNT(IUNIT,NEL,CRNT)
C
C Call for element heading.
C
       CALL ELFFLD(PHIPP,THETPP,JFLAG,IUNIT,ETA,GAM,EFX,EFY,EFZ)
       JFLAG=1
      ENDIF
C
C Reset array electric fields to zero.
C
      AEFX=DCMPLX(0.,0.)
      AEFY=DCMPLX(0.,0.)
      AEFZ=DCMPLX(0.,0.)
C
C Determine cosines and sines of angles.
C
      CTH=DCOS(THETA)
      STH=DSIN(THETA)
      CPH=DCOS(PHI)
      SPH=DSIN(PHI)
C
C Determine coordinates of point in space at which electric field is
C being calculated.
C
      X=R*CPH*STH
      Y=R*SPH*STH
      Z=R*CTH
C
C Determine electric field for given values of phi and theta.
C
 200  DO K=1,NEL
       IF(R.NE.0. .OR. K.EQ.1) THEN
C
C Determine angles with respect each element terminal.
C
        PPP=DSQRT((X-XP)*(X-XP)+(Y-YP(K))*(Y-YP(K)))
        RPP=DSQRT(PPP*PPP+(Z-ZP)*(Z-ZP))
        IF(R.NE.0.) THEN
          PHIPP=PI/2.
          IF(X.NE.XP) PHIPP=DATAN2((Y-YP(K)),(X-XP))
          THETPP=PI/2.
          IF(Z.NE.ZP) THETPP=DATAN(PPP/(Z-ZP))
         ELSE
          PHIPP=PHI
          THETPP=THETA
        ENDIF
C
C Calculate element's electric field.
C
```

```
          CALL ELFFLD(PHIPP,THETPP,JFLAG,IUNIT,ETA,GAM,EFX,EFY,EFZ)
         ENDIF
C
C Multiply by current and range dependence.
C
         IF(R.EQ.0.) THEN
           PTHDIF=CDEXP(GAM*YP(K)*STH*SPH)
           EFLDX=PTHDIF*CRNT(K)*EFX
           EFLDY=PTHDIF*CRNT(K)*EFY
           EFLDZ=PTHDIF*CRNT(K)*EFZ
          ELSE
           RNGFTR=(CDEXP(-GAM*RPP))/RPP
           EFLDX=RNGFTR*CRNT(K)*EFX
           EFLDY=RNGFTR*CRNT(K)*EFY
           EFLDZ=RNGFTR*CRNT(K)*EFZ
         ENDIF
C
C Add all contributions to electric field.
C
         AEFX=AEFX+EFLDX
         AEFY=AEFY+EFLDY
         AEFZ=AEFZ+EFLDZ
        ENDDO
C
C Return to the main program.
C
        RETURN
C
C Format statements.
C
 101    FORMAT('0',A30)
 1001 FORMAT(1X,'RANGE = ',F6.0,' kilometers')
 1002 FORMAT(1X,'FAR FIELD CALCULATION')
C
        END
C************************************************************************
```

```
C+
      SUBROUTINE ELPOS(NAME,NEL,XP,YP,ZP)
C
C *******************************************************************
C *                                                                 *
C *    SUBROUTINE ELPOS_RCVR                                        *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:      DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:        01-AUGUST-1988
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.ARRAY]ELPOS_RCVR.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED: NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELPOS_RCVR
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,ELPOS_RCVR,
C                                       'CURRNT','ELFFLD'
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the element positions of the
C  specified linear receiver antenna array and returns them to the main
C  program.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  READ
C               - array name
C               - array indicator
C               - # of elements
C               - element spacing (inches)
C
C       2.  DO
C               for each element
C
C                   Determine position of that element.
C
C           ENDDO
C
C       3.  DO
C
C               for each element
C
C                   Translate coordinates so center of array is located
C                   at origin.
C
C           ENDDO
C
C       4.  RETURN
C
C
C  INPUTS   EXPLICIT:
C       FOR002 - ELPOS_RCVR.INP
C               NAME   - name of array
```

```
C               IARRAY  - array indicator
C                         IARRAY = 11 : single element
C                                  12 : 400 ft. receiver antenna
C                                  13 : 600 ft. receiver antenna
C                                  14 : 1200 ft. receiver antenna
C                                  15 : 2400 ft. receiver antenna
C
C       FOR011 - ARRAY_SRE.INP  \     /
C       FOR012 - ARRAY_400.INP  |     | ELSPG   - element spacing
C       FOR013 - ARRAY_600.INP  |_____|           within bay (inches)
C       FOR014 - ARRAY_1200.INP |     | NEL     - # of elements per bay
C       FOR015 - ARRAY_2400.INP /     \
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C               NAME     - name of array
C               NEL      - # of elements in array
C               XP       - x position of all elements
C               YP(L)    - y position of element L
C               ZP       - z position of all elements
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C
C       CFIM             - conversion factor: inches to meters
C       NEB              - # of elements per bay for DO loop
C
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      CHARACTER*30 NAME
      DIMENSION YP(2592)
C
C Define constants.
C
      CFIM=.3048D0/12.D0
C
C Read specified array and array indicator.
C
      READ(2,100) NAME
      READ(2,*) IARRAY
C
C Read input data from specified array data file.
C
      READ(IARRAY,*) NEL
      READ(IARRAY,*) ELSPG
C
C Set initial conditions.
C
      L=0
      C=0.
C
C Calculate y coordinate of each element.
```

```
C
          DO L=1,NEL
           YP(L)=(L-1)*ZLSPG*CFIM
          ENDDO
C
C Place origin in center of array.
C
      DO L=1,NEL
       YP(L)=YP(L)-YP(NEL)/2.
      ENDDO
C
C All elements have same x and z coordinates.
C
      XP=0.
      ZP=0.
C
 100  FORMAT(A)
C
      RETURN
      END
C************************************************************************
```

```
C+
      SUBROUTINE ELPOS(NAME,NEL,XP,YP,ZP)
C
C **********************************************************************
C *                                                                    *
C *    SUBROUTINE ELPOS_TRNS                                           *
C *                                                                    *
C **********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         15-MARCH-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELPOS_TRNS.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELPOS_TRNS
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN', ARRAY_SBR,ELPOS_TRNS,
C                                       'CURRENT','ELFFLD'
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the element positions of the
C  specified linear array and returns them to the main program.
C  The coordinate system has the positive x, y, and z aligned with the
C  directions west, south, and vertical, respectively.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  READ
C                - array name
C                - array indicator
C                - # of bays
C                - # of elements per bay
C                - # of elements per bay in bay # 8 (if applicable)
C                - element spacing
C                - bay spacing
C                - road gap (if applicable)
C
C       2.  DO
C                for each bay
C
C                    Determine the number of elements in that bay.
C
C                    DO
C                        for each element
C
C                            Determine position of that element.
C
C                    ENDDO
C
C                    Add spacing to first dipole of next bay.
C
C           ENDDO
C
C       3.  DO
```

```
C
C              for each element
C
C                  Translate coordinates so center of array is located
C                  at origin.
C
C          ENDDO
C
C      4.  RETURN
C
C
C  INPUTS   EXPLICIT:
C      FOR002 - ELPOS_TRNS.INP
C              NAME     - name of array
C              IARRAY   - array indicator
C                         IARRAY = 11 : single transmitter element
C                                  12 : single Kickapoo bay
C                                  13 : Jordan Lake
C                                  14 : Gila River
C                                  15 : Kickapoo Complex
C                                  16 : North Kickapoo
C                                  17 : South Kickapoo
C                                  18 : variable transmitter array
C                                     / BYSPG    - bay spacing, in
C                                     |            excess of element
C                                     |            spacing (inches)
C                                     | ELSPG    - element spacing
C      FOR011 - ARRAY_STE.INP  \      |            within bay (inches)
C      FOR012 - ARRAY_KB.INP   |      | NBAY     - # of bays
C      FOR013 - ARRAY_JL.INP   |      | NELBY    - normal # of elements
C      FOR014 - ARRAY_GR.INP   |______|            per bay
C      FOR015 - ARRAY_KC.INP   |      | NELBY8   - # of elements in bay
C      FOR016 - ARRAY_KN.INP   |      |            #8 for North Kickapoo
C      FOR017 - ARRAY_KS.INP   |      |            Bay #8 is split by a
C      FOR018 - ARRAY_VTA.INP  /      |            road
C                                     | RDGAP    - road gap within bay
C                                     |            #8, in excess of
C                                     |            element spacing, for
C                                     |            North Kickapoo
C                                     \            (inches)
C
C          IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C              NAME      - name of array
C              NEL       - # of elements in array
C              XP        - x position of all elements
C              YP(L)     - y position of element L
C              ZP        - z position of all elements
C
C          IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C
C      CFIM             - conversion factor: inches to meters
C      NEB              - # of elements per bay for DO loop
C
C
C  SYSTEM STATE CHANGES:  NONE
C
```

```
C-
      IMPLICIT REAL*8(A-H,O-Z)
      CHARACTER*30 NAME
      DIMENSION YP(2592)
C
      CFIM=.3048D0/12.D0
C
C Read specified array and array indicator.
C
      READ(2,100) NAME
      READ(2,*) IARRAY
C
C Read input data from specified array data file.
C
      READ(IARRAY,*) NBAY
      READ(IARRAY,*) NELBY
      READ(IARRAY,*) NELBY8
      READ(IARRAY,*) ELSPG
      READ(IARRAY,*) BYSPG
      READ(IARRAY,*) RDGAP
C
C Set initial conditions.
C
      L=0
      C=0.
C
C Calculate y coordinate of each element.
C
      DO I=1,NBAY
       IF(I.NE.8 .OR. IARRAY.EQ.17) THEN
         NEB=NELBY
         DO J=1,NEB
          L=L+1
          YP(L)=(L-1)*ELSPG*CFIM+C
         ENDDO
         C=C+BYSPG*CFIM
        ELSE
         NEB=NELBY8/2.
         DO K=1,2
          DO J=1,NEB
           L=L+1
           YP(L)=(L-1)*ELSPG*CFIM+C
          ENDDO
          IF(K.EQ.1) THEN
            C=C+RDGAP*CFIM
           ELSE
            C=C+BYSPG*CFIM
          ENDIF
         ENDDO
       ENDIF
      ENDDO
C
C Detemine total of elements for array.
C
      NEL=L
C
C Place origin in center of array.
C
      DO L=1,NEL
       YP(L)=YP(L)-YP(NEL)/2.
```

```
      ENDDO
C
C All elements have same x and z coordinates.
C
      XP=0.
      ZP=0.
C
 100  FORMAT(A)
C
      RETURN
      END
C*************************************************************************
```

```
C+
      SUBROUTINE CURRNT(IUNIT,NEL,CRNT)
C
C *******************************************************************
C *                                                                 *
C *    SUBROUTINE CURRNT_RBAY                                       *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         14-JUNE-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]CURRNT_RBAY.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  RANDG  - generates a single value of a set of
C                                random numbers which falls within a
C                                Gaussian distribution of specified
C                                mean value and width.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN CURRNT_RBAY
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       CURRNT_RBAY,'ELFFLD'
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates a set of random current amplitude and
C  phase values with a Gaussian distribution of specified mean value and
C  width for the set of 18 Kickapoo complex bays.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. READ
C               - mean current at array element terminal for a specific
C                 all bays
C               - current amplitude sigma
C               - current phase sigma
C
C       2. WRITE
C               - mean current at array element terminal for a specific
C                 all bays
C               - current amplitude sigma
C               - current phase sigma
C
C       3. DO
C               for each bay
C
C                   CALL
C                       - random number for current amplitude within
C                         specified Gaussian distribution.
C                       - random number for current amplitude within
C                         specified Gaussian distribution.
C
C                   DO
C                       for each element
C
C                           Assign complex current value.
```

```
C
C                   ENDDO
C
C                   WRITE
C                       - bay #
C                       - current amplitude
C                       - current phase
C
C          ENDDO
C
C       4. RETURN
C
C
C  INPUTS
C    EXPLICIT:
C       IUNIT   - output FORTRAN unit number
C       NEL     - # of radiating elements
C
C       FOR008 - CURRNT_RBAY.INP
C               AMPSIG  - Gaussian value of sigma for current amplitudes
C               CTMEAN  - mean maximum current value at array elements'
C                         terminals (amps)
C               PHSSIG  - Gaussian value of sigma for current phases
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       CRNT(K) - complex current value for element K
C
C       FOR'IUNIT' - 'MAIN'.OUT
C               AMPSIG  - Gaussian value of sigma for current amplitudes
C                         (amperes)
C               CTMEAN  - mean maximum current value at array elements'
C                         terminals (amperes)
C               PHSSIG  - Gaussian value of sigma for current phases
C                         (degrees)
C          For each bay.
C               I       - bay #
C               CTAMP   - current phase at array element terminal
C                         (amperes)
C               CTPHSD  - current amplitude at array element terminal
C                         (degrees)
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C       CTPHS   - current amplitude at array element terminal (radians)
C       IX1     - seed value for current amplitude
C       IY1     - seed value for current amplitude
C       IX2     - seed value for current phase
C       IY2     - seed value for current phase
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
```

```
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CIM
      COMPLEX*16 CRNT(2556)
C
C  Define constants.
C
      PI=3.141592653589793
      CIM=DCMPLX(0.,1.)
      CFDR=PI/180.
C
C  Read input values.
C
      READ(8,*) CTMEAN
      READ(8,*) AMPSIG
      READ(8,*) PHSSIG
C
C  Write input values.
C
      WRITE(IUNIT,1003) CTMEAN
      WRITE(IUNIT,1004) AMPSIG
      WRITE(IUNIT,1005) PHSSIG
C
      PHSSIG=PHSSIG*CFDR
C
C  Set random generator seed values.
C
      IX1=0
      IY1=0
      IX2=1
      IY2=1
C
C  Use random generator to determine current values of each bay.
C
      L=0
      DO I=1,18
       CALL RANDG(IX1,IY1,AMPSIG,CTMEAN,CTAMP)
       CALL RANDG(IX2,IY2,PHSSIG,0.,CTPHS)
       IF(I.NE.8) THEN
         DO J=1,144
          L=L+1
          CRNT(L)=CTAMP*CDEXP(CIM*CTPHS)
         ENDDO
        ELSE
         DO K=1,2
          DO J=1,54
           L=L+1
           CRNT(L)=CTAMP*CDEXP(CIM*CTPHS)
          ENDDO
         ENDDO
       ENDIF
       CTPHSD=CTPHS/CFDR
       WRITE(IUNIT,1006) I,CTAMP,CTPHSD
      ENDDO
C
C Format statements.
C
 1003 FORMAT('0','MEAN CURRENT = 'F5.3' amps')
 1004 FORMAT(1X,'AMPLITUDE SIGMA = 'F5.3' amps')
 1005 FORMAT(1X,'PHASE SIGMA = 'F5.3' degrees')
 1006 FORMAT(1X,'BAY ',I2,' CURRENT AMPLITUDE : ',F4.2,' amps
```

```
     +CURRENT PHASE : ',F8.3,' degrees')
      RETURN
      END
      INCLUDE '[BERG.ARRAY]RANDG.FOR/LIST'
C***********************************************************************
```

```
C+
      SUBROUTINE CURRNT(IUNIT,NEL,CRNT)
C
C **********************************************************************
C *                                                                    *
C *    SUBROUTINE CURRNT_RSBY                                          *
C *                                                                    *
C **********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         06-OCTOBER-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]CURRNT_RSBY.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN CURRNT_RSBY
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN', ARRAY_SBR,'ELPOS',
C                                       CURRNT_RSBY,'ELFFLD'
C
C  PARENT PROGRAM:      ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine returns a complex current value for each dipole
C  in the Kickapoo Complex from an input set of average bay amplitude
C  and phase values.  The dipoles' current phases are randomly
C  distributed about the average bay value.  The distribution is
C  Gaussian.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  READ
C                - Gaussian value of sigma for current amplitudes
C                - Gaussian value of sigma for current phases
C
C       2.  WRITE
C                - Gaussian value of sigma for current amplitudes
C                - Gaussian value of sigma for current phases
C
C       3.  DO
C                for each bay
C
C                    READ
C                        - average current amplitude.
C                        - average current phase.
C           ENDDO
C
C       4.  DO
C                for each bay
C
C                    Set random number generator seeds.
C
C           ENDDO
C
C       5.  DO
C                for each bay
C
```

```
C                    DO
C                        for each element
C
C                            CALL
C                                random number generator
C
C                            Calculate complex current.
C
C                    ENDDO
C
C            ENDDO
C
C        6.  RETURN
C
C  INPUTS
C    EXPLICIT:
C       IUNIT    - output FORTRAN unit number
C       NEL      - # of radiating elements
C
C       FOR008 - CURRNT_RSBY.INP
C               AMPSIG          - Gaussian value of sigma for current
C                                 amplitudes
C               PHSSIG          - Gaussian value of sigma for current
C                                 phases
C               BCTAMP(I)       - average current amplitude at array
C                                 element terminal for bay I
C               BCTPHS(I)       - average current phase at array element
C                                 terminal for bay I
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       CRNT(K) - complex current value for element K
C
C       FOR'IUNIT' - CURRNT_RSBY.INP
C               AMPSIG          - Gaussian value of sigma for current
C                                 amplitudes
C               PHSSIG          - Gaussian value of sigma for current
C                                 phases
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C       CFDR     - conversion factor: degrees to radians
C       CTAMP    - current phase at array element terminal
C       CTPHS    - current amplitude at array element terminal
C       IX1(I)   - seed value for current amplitude for bay I
C       IY1(I)   - seed value for current amplitude for bay I
C       IX2(I)   - seed value for current phase for bay I
C       IY2(I)   - seed value for current phase for bay I
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
```

```
      COMPLEX*16 CIM
      COMPLEX*16 CRNT(2556)
      DIMENSION IX1(18),IX2(18),IY1(18),IY2(18),BCTAMP(18),BCTPHS(18)
C
C  Define constants.
C
      PI=3.141592653589793
      PI2=PI/2.
      CIM=DCMPLX(0.,1.)
      CFDR=PI/180.
C
C  Read input values.
C
      READ(8,*) AMPSIG
      READ(8,*) PHSSIG
C
C  Write input values.
C
      WRITE(IUNIT,1004) AMPSIG
      WRITE(IUNIT,1005) PHSSIG
C
      PHSSIG=PHSSIG*CFDR
C
C  Read dipole current values for each element in each bay.
C  Convert phase from degrees to radians
C
      DO I=1,18
       READ(8,*) BCTAMP(I),BCTPHS(I)
       BCTPHS(I) = BCTPHS(I) * CFDR
      ENDDO
C
C  Set random generator seed values.
C
      DO I=1,18
       IX1(I)=I
       IY1(I)=I+1
       IX2(I)=2*I
       IY2(I)=2*I+1
      ENDDO
C
C  Use random generator to determine current values of each element.
C
      L=0
      DO I=1,18
       IF(I.NE.8) THEN
         DO J=1,144
          L=L+1
          CALL RANDG(IX1(I),IY1(I),AMPSIG,BCTAMP(I),CTAMP)
          CALL RANDG(IX2(I),IY2(I),PHSSIG,BCTPHS(I),CTPHS)
          CRNT(L)=CTAMP*CDEXP(CIM*CTPHS)
         ENDDO
        ELSE
         DO K=1,2
          DO J=1,54
           L=L+1
           CALL RANDG(IX1(I),IY1(I),AMPSIG,BCTAMP(8),CTAMP)
           CALL RANDG(IX2(I),IY2(I),PHSSIG,BCTPHS(8),CTPHS)
           CRNT(L)=CTAMP*CDEXP(CIM*CTPHS)
          ENDDO
         ENDDO
```

```
        ENDIF
       ENDDO
C
C  Format statements.
C
 1003 FORMAT('0','MEAN CURRENT = 'F5.3' amps')
 1004 FORMAT(1X,'AMPLITUDE SIGMA = 'F5.3' amps')
 1005 FORMAT(1X,'PHASE SIGMA = 'F5.3' degrees')
      RETURN
      END
      INCLUDE '[BERG.ARRAY]RANDG.FOR/LIST'
C************************************************************************
```

```
C+
      SUBROUTINE CURRNT(IUNIT,NEL,CRNT)
C
C ***********************************************************************
C *                                                                     *
C *     SUBROUTINE CURRNT_SBAY                                          *
C *                                                                     *
C ***********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         04-OCTOBER-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]CURRNT_SBAY.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:          $ FORTRAN CURRNT_SBAY
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN', ARRAY_SBR,'ELPOS',
C                                       CURRNT_SBAY,'ELFFLD'
C
C  PARENT PROGRAM:      ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine returns a complex current value for each dipole
C  in the Kickapoo Complex from an input set of average bay amplitude
C  and phase values.  The dipoles in each bay have the same complex
C  current as the bay average.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. DO
C               for each bay
C
C                   READ
C                       - current amplitude of each dipole
C                       - current phase of each dipole
C
C          ENDDO
C
C       2. DO
C               for each element
C
C                   - assign complex current value
C
C          ENDDO
C
C       3. RETURN
C
C
C  INPUTS
C    EXPLICIT:
C       IUNIT   - output FORTRAN unit number
C       NEL     - # of radiating elements
C
C       FOR008 - CURRNT_SBAY.INP
C               CTAMP(I)       - current amplitude of each dipole in
C                                bay I (amperes)
```

```
C               CTPHS(I)        - current phase of each dipole in bay I
C                                 (degrees)
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C      CRNT(K) - complex current value for element K
C
C      FOR'IUNIT' - CURRNT_SBAY.INP
C               I               - bay #
C               CTAMP(I)        - current amplitude of each dipole in
C                                 bay I (amperes)
C               CTPHSD          - bay phase in degrees
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C      CFDR    - conversion factor: degrees to radians
C      CTPHSD  - bay phase in degrees
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CIM
      COMPLEX*16 CRNT(2556)
      DIMENSION CTPHS(18),CTAMP(18)
C
C  Define constants.
C
      PI = 3.141592653589793
      CIM = DCMPLX(0.,1.)
      CFDR = PI/180.
C
C  Read dipole current values for each element in each bay.
C  Convert phase from degrees to radians
C
      DO I=1,18
       READ(8,*) CTAMP(I),CTPHS(I)
       CTPHS(I) = CTPHS(I) * CFDR
      ENDDO
C
C  Assign complex current value to each dipole in each bay.
C
      L=0
      DO I=1,18
       IF(I.NE.8) THEN
         DO J=1,144
          L=L+1
          CRNT(L) = CTAMP(I) * CDEXP(CIM*CTPHS(I))
         ENDDO
        ELSE
         DO K=1,2
          DO J=1,54
           L=L+1
```

```
          CRNT(L) = CTAMP(I) * CDEXP(CIM*CTPHS(I))
         ENDDO
        ENDDO
      ENDIF
C
C  Convert bay phase to degrees.
C
      CTPHSD=CTPHS(I)/CFDR
C
C  Write dipoles' phase and amplitude for each bay.
C
      WRITE(IUNIT,1006) I,CTAMP(I),CTPHSD
     ENDDO
C
 1003 FORMAT('0','MEAN CURRENT = 'F5.3' amps')
 1004 FORMAT(1X,'AMPLITUDE SIGMA = 'F5.3' amps')
 1005 FORMAT(1X,'PHASE SIGMA = 'F5.3' degrees')
 1006 FORMAT(1X,'BAY ',I2,' CURRENT AMPLITUDE : ',F4.2,' amps
     +CURRENT PHASE : ',F8.3,' degrees')
      RETURN
      END
```

```
C+
      SUBROUTINE CURRNT(IUNIT,NEL,CRNT)
C
C *********************************************************************
C *                                                                   *
C *     SUBROUTINE CURRNT_STND                                        *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         15-MARCH-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]CURRNT_STND.FOR
C
C  CALLING ROUTINE:  @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  RANDG  - generates a single value of a set of
C                                random numbers which falls within a
C                                Gaussian distribution of specified
C                                mean value and width.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN CURRNT_STND
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       CURRNT_STND, 'ELFFLD'
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates a set of random current amplitude and
C  phase values with a Gaussian distribution of specified mean value and
C  width.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. READ
C               - mean current at array element terminal
C               - current amplitude sigma
C               - current phase sigma
C
C       2. WRITE
C               - mean current at array element terminal
C               - current amplitude sigma
C               - current phase sigma
C
C       3. DO
C               for each element
C
C                   CALL
C                       - random number for current amplitude within
C                         specified Gaussian distribution.
C                       - random number for current amplitude within
C                         specified Gaussian distribution.
C          ENDDO
C
C       4. RETURN
C
C
C  INPUTS
C    EXPLICIT:
```

```
C       IUNIT   - output FORTRAN unit number
C       NEL     - # of radiating elements
C
C       FOR008 - CURRNT_STND.INP
C               AMPSIG  - Gaussian value of sigma for current amplitudes
C                         (amperes)
C               CTMEAN  - mean maximum current value at array elements'
C                         terminals (amperes)
C               PHSSIG  - Gaussian value of sigma for current phases
C                         (degrees)
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       CRNT(K) - complex current value for element K (amperes)
C
C       FOR'IUNIT' - 'MAIN'.INP
C               AMPSIG  - Gaussian value of sigma for current amplitudes
C                         (amperes)
C               CTMEAN  - mean maximum current value at array elements'
C                         terminals (amperes)
C               PHSSIG  - Gaussian value of sigma for current phases
C                         (degrees)
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C       CTAMP   - current phase at array element terminal
C       CTPHS   - current amplitude at array element terminal
C       IX1     - seed value for current amplitude
C       IY1     - seed value for current amplitude
C       IX2     - seed value for current phase
C       IY2     - seed value for current phase
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CIM
      COMPLEX*16 CRNT(2592)
C
C  Define constants.
C
      PI=3.141592653589793
      CIM=DCMPLX(0.,1.)
      CFDR=PI/180.
C
C  Read input values.
C
      READ(8,*) CTMEAN
      READ(8,*) AMPSIG
      READ(8,*) FWHM
C
C  Write input values.
C
      WRITE(IUNIT,1003) CTMEAN
      WRITE(IUNIT,1004) AMPSIG
      WRITE(IUNIT,1005) FWHM
```

```
C
C  Convert FWHM (degrees) to phase sigma (radians)
C
      PHSSIG = FWHM/2./DSQRT(2.DO * DLOG(2.DO))*CFDR
C
C  Set random generator seed values.
C
      IX1=0
      IY1=0
      IX2=1
      IY2=1
C
C  Use random generator to determine current values of each element.
C
      DO K=1,NEL
       CALL RANDG(IX1,IY1,AMPSIG,CTMEAN,CTAMP)
       CALL RANDG(IX2,IY2,PHSSIG,0.,CTPHS)
       CRNT(K)=CTAMP*CDEXP(CIM*CTPHS)
      ENDDO
C
 1003 FORMAT('0','MEAN CURRENT = 'F5.3' amps')
 1004 FORMAT(1X,'AMPLITUDE SIGMA = 'F5.3' amps')
 1005 FORMAT(1X,'PHASE FWHM = 'F7.3' degrees')
      RETURN
      END
      INCLUDE '[BERG.ARRAY]RANDG.FOR/LIST'
C***********************************************************************
```

```
                    SUBROUTINE RANDG(IX,IY,SIGMA,RMEAN,VAL)
C
C ***************************************************************************
C *                                                                         *
C *                          SUBROUTINE RANDG                               *
C *                                                                         *
C ***************************************************************************
C
C  AUTHOR:      Dr. E. James Wadiak
C  DATE:        26-JAN-1988
C  LANGUAGE:    FORTRAN ANSI-77  (VAX/VMS operating system)
C  FILE:        VX7770::SPACE:[WADIAK.LSQ.SIM]RANDG.FOR
C               MV7770::SPACE:[BERG.ARRAY]RANDG.FOR
C
C  CALLING ROUTINES:             ERROR.FOR
C                                MULTLSQ.FOR
C                                CURRNT_STND.FOR
C
C  SUBROUTINES CALLED:           RANDU - VAX library subroutine which returns
C                                        a random number uniformly distributed
C                                        between 0 and 1.
C
C  COMPILE INSTRUCTIONS:         via INCLUDE statement in parent program.
C
C  LINK/LOAD INSTRUCTIONS:       via INCLUDE statement in parent program.
C
C  PARENT PROGRAMS:              SIMDAT.FOR, MULTLSQ.FOR
C                                CURRNT_STND.FOR
C
C  PROGRAM DESCRIPTION:
C
C     This subroutine applies the Central Limit Theorem to derive a random
C  number VAL whose distribution is Gaussian with a characteristic dispersion
C  of SIGMA and a mean value of RMEAN.
C
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C    1. Sum 12 random numbers uniformly distributed between 0 and 1.  The
C       resultant number is Gaussian-distributed about the expection value of
C       <6> with a standard deviation of 1.
C
C    2. Multiply the deviation from the expectation value times the desired
C       standard deviation, and add the desired mean.  This produces a random
C       number with the desired properties.
C
C    3. RETURN to the calling program.
C
C  INPUTS   EXPLICIT (via arguments to the CALL statement):
C
C               IX - random number generator seed.
C               IY - random number generator seed.
C               RMEAN - desired mean of the output random number.
C               SIGMA - desired standard deviation of the output random number.
C
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT (via the arguments to the CALL statement):
C
```

```
C                IX - new seed for next call to RANDG.
C                IY - new seed for next call to RANDG.
C                VAL - random number with the desired distribution properties.
C
C
C            IMPLICIT:   NONE
C
C  MAJOR VARIABLES:
C
C                A - sum of the 12 uniformly distributed random numbers.
C                RMEAN - desired mean of output random number.
C                SIGMA - desired standard deviation of output random number.
C                VAL - output random number with desired properties.
C
C
C  MODIFIED:
C
C
C*******************************************************************************
        IMPLICIT REAL*8 (A-H,O-Z)
        REAL*4 Y
        A = 0.0
        DO I=1,12
          CALL RANDU(IX,IY,Y)
          A = A + Y
        ENDDO
        VAL = ( A - 6.0 ) * SIGMA + RMEAN
        RETURN
        END
```

```
C+
      SUBROUTINE ELFFLD(PHI,THETA,IFLAG,IUNIT,ETA,GAMMA,EFX,EFY,EFZ)
C
C *******************************************************************
C *                                                                 *
C *    SUBROUTINE ELFFLD_AHGP                                       *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         23-NOVEMBER-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELPOS_AHGP
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELFFLD_AHGP
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       'CURRNT',ELFFLD_AHGP
C
C  PARENT PROGRAM:  ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the polar components of the range
C  independent far electric field for the NAVSPASUR half-wave arrowhead
C  dipole transmitter element over an infinite ground screen.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate the far electric field for a specified angle.
C
C       2. RETURN
C
C
C  INPUTS
C    EXPLICIT:
C       ETA      - intrinsic impedance of ambient medium
C       IFLAG    - flag : 0 => first time called
C                         1 => called previously
C       IUNIT    - output unit number
C       PHI      - angle phi (radians) wrt element
C       THETA    - angle theta (radians) wrt element
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EFX      - x component of far electric field with range and
C                  current dependence suppressed.
C       EFY      - x component of far electric field with range and
C                  current dependence suppressed.
C       EFZ      - x component of far electric field with range and
C                  current dependence suppressed.
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
```

```
C
C       ANGFTRN - primary angle dependent term with negative term
C       ANGFTRP - primary angle dependent term with positive term
C       CIM     - complex i
C       EFO     - constant factor
C       EFP     - range independent phi component of the far electric
C                 field
C       EFT     - range independent theta component of the far electric
C                 field
C       EXPNN   - nondirectional term with two negative factors
C       EXPNP   - nondirectional term with negative and positive factors
C       EXPPN   - nondirectional term with positive and negative factors
C       EXPPP   - nondirectional term with two positive factors
C       IMGFTR  - image dipole factor
C       PHIFTRN - phi component factor with negative term
C       PHIFTRP - phi component factor with positive term
C       THTFTRN - theta component factor with negative term
C       THTFTRP - theta component factor with positive term
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,GAMMA,EFT,EFP,EFX,EFY,EFZ,EFO
      COMPLEX*16 CIM,EXPNN,EXPNP,EXPPP,EXPPN,IMGFTR
C
C Define constants.
C
      DATA EO,UO/8.854187935760775 9D-12,1.2566370614355172D-6/
      DATA PI/3.141592653589793/
      PI2=PI/2.
      TWOPI=2.*PI
      CIM=DCMPLX(0.,1.)
      CFIM=.3048/12.
      CFDR=PI/180.
      OMEGA=2.*PI*216.980*10.**6
      GAMMA=OMEGA*DSQRT(UO*EO)*CIM
      ETA=DSQRT(UO/EO)
      DELTA=55.*CFDR
      EFO=CIM*ETA/2./TWOPI
C
C Calculate sines and cosines.
C
      CSPHI=DCOS(PHI)
      SNPHI=DSIN(PHI)
      CSTHET=DCOS(THETA)
      SNTHET=DSIN(THETA)
      CSDELT=DCOS(DELTA)
      SNDELT=DSIN(DELTA)
C
C Adjust for precision errors.
C
      IF(DABS(CSPHI).LT.1.D-15) CSPHI=0.
      IF(DABS(SNPHI).LT.1.D-15) SNPHI=0.
      IF(DABS(CSTHET).LT.1.D-15) CSTHET=0.
      IF(DABS(SNTHET).LT.1.D-15) SNTHET=0.
C
```

```
C Calculate angle dependent terms.
C
      ANGFTRP=SNDELT*CSTHET+CSDELT*SNTHET*CSPHI
      ANGFTRN=SNDELT*CSTHET-CSDELT*SNTHET*CSPHI
C
C Calculate nondirectional terms.
C
      EXPNN=CDEXP(-CIM*PI2*ANGFTRN)
      EXPNP=CDEXP(-CIM*PI2*ANGFTRP)
      EXPPP=CDEXP(CIM*PI2*ANGFTRP)
      EXPPN=CDEXP(CIM*PI2*ANGFTRN)
C
C Calculate directional component factors.
C
      IF(ANGFTRP.NE.1.) THEN
        THTFTRP=(SNDELT*SNTHET-CSDELT*CSTHET*CSPHI)/(1.-(ANGFTRP)**2.)
        PHIFTRP=CSDELT*SNPHI/(1.-(ANGFTRP)**2.)
       ELSE
        THTFTRP=1.
        PHIFTRP=1.
      ENDIF
      IF(ANGFTRN.NE.1.) THEN
        THTFTRN=-(SNDELT*SNTHET+CSDELT*CSTHET*CSPHI)/(1.-(ANGFTRN)**2.)
        PHIFTRN=CSDELT*SNPHI/(1.-(ANGFTRN)**2.)
       ELSE
        THTFTRN=-1.
        PHIFTRN=1.
      ENDIF
C
C Calculate image factor.
C
      IMGFTR=CDEXP(-CIM*3.*PI2*CSTHET)
C
C Calculate far electric field components.
C
      EFT=EFO*((EXPNP+CIM*ANGFTRP)*THTFTRP
     +   +     (EXPNN+CIM*ANGFTRN)*THTFTRN
     +   -     (EXPPN-CIM*ANGFTRN)*IMGFTR*THTFTRN
     +   -     (EXPPP-CIM*ANGFTRP)*IMGFTR*THTFTRP)
      EFP=EFO*((EXPNP+CIM*ANGFTRP)*PHIFTRP
     +   +     (EXPNN+CIM*ANGFTRN)*PHIFTRN
     +   -     (EXPPN-CIM*ANGFTRN)*IMGFTR*PHIFTRN
     +   -     (EXPPP-CIM*ANGFTRP)*IMGFTR*PHIFTRP)
C
C Convert spherical components to cartesian components.
C
      EFX=EFT*CSTHET*CSPHI-EFP*SNPHI
      EFY=EFT*CSTHET*SNPHI+EFP*CSPHI
      EFZ=-EFT*SNTHET
C
C Return to subroutine.
C
      RETURN
      END
C***********************************************************************
```

```
C+
      SUBROUTINE ELFFLD(PHI,THETA,IFLAG,IUNIT,ETA,GAMMA,EFX,EFY,EFZ)
C
C ********************************************************************
C *                                                                  *
C *    SUBROUTINE ELFFLD_AHHW                                        *
C *                                                                  *
C ********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         23-NOVEMBER-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELPOS_AHHW
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELFFLD_AHHW
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       'CURRNT',ELFFLD_AHHW
C
C  PARENT PROGRAM:  ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the polar components of the range
C  independent far electric field for an half-wave arrowhead dipole
C  NAVSPASUR transmitter element in free space.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate the far electric field for a specified angle.
C
C       2. RETURN
C
C  INPUTS
C    EXPLICIT:
C       ETA      - intrinsic impedance of ambient medium
C       IFLAG    - flag : 0 => first time called
C                         1 => called previously
C       IUNIT    - output unit number
C       PHI      - angle phi (radians) wrt element
C       THETA    - angle theta (radians) wrt element
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EFX      - x component of far electric field with range and
C                  current dependence suppressed.
C       EFY      - x component of far electric field with range and
C                  current dependence suppressed.
C       EFZ      - x component of far electric field with range and
C                  current dependence suppressed.
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C       ANGFTRN - primary angle dependent term with negative term
```

```
C       ANGFTRP - primary angle dependent term with positive term
C       CIM     - complex i
C       EFO     - constant factor
C       EFP     - range independent phi component of the far electric
C                 field
C       EFT     - range independent theta component of the far electric
C                 field
C       EXPNN   - nondirectional term with two negative factors
C       EXPNP   - nondirectional term with negative and positive factors
C       PHIFTRN - phi component factor with negative term
C       PHIFTRP - phi component factor with positive term
C       THTFTRN - theta component factor with negative term
C       THTFTRP - theta component factor with positive term
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,GAMMA,EFT,EFP,EFX,EFY,EFZ,EFO
      COMPLEX*16 CIM,EXPNN,EXPNP
C
C Define constants.
C
      DATA EO,UO/8.8541879357607759D-12,1.2566370614355172D-6/
      DATA PI/3.141592653589793/
      PI2=PI/2.
      TWOPI=2.*PI
      CIM=DCMPLX(0.,1.)
      CFIM=.3048/12.
      CFDR=PI/180.
      OMEGA=2.*PI*216.980*10.**6
      GAMMA=OMEGA*DSQRT(UO*EO)*CIM
      ETA=DSQRT(UO/EO)
      DELTA=55.*CFDR
      EFO=CIM*ETA/2./TWOPI
C
C Calculate sines and cosines.
C
      CSPHI=DCOS(PHI)
      SNPHI=DSIN(PHI)
      CSTHET=DCOS(THETA)
      SNTHET=DSIN(THETA)
      CSDELT=DCOS(DELTA)
      SNDELT=DSIN(DELTA)
C
C Adjust for precision errors.
C
      IF(DABS(CSPHI).LT.1.D-15) CSPHI=0.
      IF(DABS(SNPHI).LT.1.D-15) SNPHI=0.
      IF(DABS(CSTHET).LT.1.D-15) CSTHET=0.
      IF(DABS(SNTHET).LT.1.D-15) SNTHET=0.
C
C Calculate angle dependent terms.
C
      ANGFTRP=SNDELT*CSTHET+CSDELT*SNTHET*CSPHI
      ANGFTRN=SNDELT*CSTHET-CSDELT*SNTHET*CSPHI
C
```

```
C Calculate nondirectional terms.
C
      EXPNN=CDEXP(-CIM*PI2*ANGFTRN)
      EXPNP=CDEXP(-CIM*PI2*ANGFTRP)
C
C Calculate directional component factors.
C
      IF(ANGFTRP.NE.1.)THEN
        THTFTRP=(SNDELT*SNTHET-CSDELT*CSTHET*CSPHI)/(1.-ANGFTRP**2.)
        PHIFTRP=CSDELT*SNPHI/(1.-ANGFTRP**2.)
       ELSE
        THTFTRP=1.
        PHIFTRP=1.
      ENDIF
      IF(ANGFTRN.NE.1.)THEN
        THTFTRN=-(SNDELT*SNTHET+CSDELT*CSTHET*CSPHI)/(1.-ANGFTRN**2.)
        PHIFTRN=CSDELT*SNPHI/(1.-ANGFTRN**2.)
       ELSE
        THTFTRN=-1.
        PHIFTRN=1.
      ENDIF
C
C Calculate far electric field components.
C
      EFT=EFO*((EXPNP+CIM*ANGFTRP)*THTFTRP
     +   +     (EXPNN+CIM*ANGFTRN)*THTFTRN)
      EFP=EFO*((EXPNP+CIM*ANGFTRP)*PHIFTRP
     +   +     (EXPNN+CIM*ANGFTRN)*PHIFTRN)
C
C  Convert spherical components to cartesian components.
C
      EFX=EFT*CSTHET*CSPHI-EFP*SNPHI
      EFY=EFT*CSTHET*SNPHI+EFP*CSPHI
      EFZ=-EFT*SNTHET
C
      RETURN
      END
C************************************************************************
```

```
C+
      SUBROUTINE ELFFLD(PHI,THETA,IFLAG,IUNIT,ETA,GAM,EFX,EFY,EFZ)
C
C **********************************************************************
C *                                                                    *
C *      SUBROUTINE ELFFLD_APRX                                        *
C *                                                                    *
C **********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         18-MAY-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELFFLD_APRX.FOR
C
C  CALLING ROUTINE:      @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:          $ FORTRAN ELFFLD_APRX
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                      'CURRNT',ELFFLD_APRX
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine uses a polynomial fit to the actual pattern to
C  calculate the range independent far electric field for a specified
C  direction of a NAVSPASUR arrowhead dipole with unit terminal current
C  placed over a finite thin wire-grid ground screen.  This
C  approximation is only good for angles of three degrees or less off
C  the east-west plane.
C       In the region within three degrees of the east-west plane, each
C  component of the electric field can be split into east-west factor
C  and a north-south factor.  For the x and z components the north-south
C  factor is one, the value of the components has no north-south
C  dependence. However, the y components has both an east-west and a
C  north-south dependence.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  IF
C                this is the first time the subroutine is called
C
C                    WRITE
C                        - Element name
C
C           ENDIF
C
C       2.  Convert coordinates to the coordinates of polynomial.
C
C       3.  Calculate cartesian components of total electric field.
C
C       4.  RETURN
C
C  INPUTS
C    EXPLICIT:
C       IFLAG    - flag : 0 => first time called
C                         1 => called previouly
C       IUNIT    - ouput FOTRAN unit number
```

```
C        PHI      - standard spherical coordinate phi (radians)
C        THETA    - standard spherical coordinate theta (radians)
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C        EFX      - x component of far electric field from element with
C                   range and current dependence suppressed (mks units)
C        EFY      - y component of far electric field from element with
C                   range and current dependence suppressed (mks units)
C        EFZ      - z component of far electric field from element with
C                   range and current dependence suppressed (mks units)
C        ETA      - intrinsic impedance of ambient medium (mks units)
C        GAM      - complex propagation constant of ambient medium (mks
C                   units)
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C        ALPHA    - projected angle onto east-west plane wrt vertical
C        BETA     - angle away from plane
C        EFXI     - imaginary part of x component of electric field
C        EFXR     - real part of x component of electric field
C        EFYEI    - imaginary part of east-west factor of y component
C                   of electric field
C        EFYER    - real value of east-west factor of y component of
C                   electric field
C        EFYNI    - imaginary part of north-south factor of y component
C                   of electric field
C        EFYNR    - real value of north-south factor of y component of
C                   electric field
C        EFZR     - imaginary part of z component of electric field
C        EFZI     - real part of z component of electric field
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 EFX,EFY,EFZ
      COMPLEX*16 ETA,GAM
      DIMENSION A(15),B(12),C(12),D(12),E(4),F(4),G(15),H(12)
C
C Set polynomial coefficents.
C
      DATA A / -48.81134500, -3.32936269, 135.92066364, -556.84675634,
     + -3415.26696738, -5076.32694090, -320.07308383, 1066.81398596,
     + -11351.79641405, -21268.31842580, -13250.76618528, -230.58635200,
     + 3613.09026635, 1667.14825305, 245.32768571 /
      DATA B / -46.24710906, 5.29885609, 289.74564581, 1550.08290961,
     + 8076.79186498, 30390.39129857, 65748.81389051, 83191.21282485,
     + 63449.26300416, 28885.86856003, 7258.09465898, 776.92366304 /
      DATA C / 0.00373171, 0.33690632, 12.14493123, 89.90107435,
     + 734.90005930, 2583.98663359, 4633.35078175, 4795.42851482,
     + 2998.85124374, 1118.81035362, 228.54026027, 19.52962161 /
      DATA D / 0.00032473, 3.93567557, 0.02370856, -48.14602902,
     + -93.74715567, -419.03535033, -1479.97883333, -2550.70395440,
```

```
     + -2385.56411205, -1260.71016825, -356.54500172, -42.15469253 /
      DATA E / 0.00017687, 57.62716404, 2.94733282, -1019.55387357 /
      DATA F / -0.00061399, 58.14454363, -6.55235737, -524.35506834 /
      DATA G / -0.02134093, -51.51615729, -75.83463372, -672.63905899,
     + -4997.52158937, -17519.54638209, -31408.47680427,
     + -26471.22647279, -2914.04969906, 11745.75285317, 6870.78830840,
     + -1398.86292882, -2758.31171124, -1034.36202167, -133.09962066 /
      DATA H / -0.04307622, -51.33941125, -122.98172857, -1088.42410730,
     + -6232.74003221, -18844.25863408, -30312.18101991,
     + -26746.27006469, -12462.69308608, -2332.73449266, 207.80311051,
     + 103.85552955 /
C
C Write element name the first time the subroutine is called.
C
      IF (IFLAG.EQ.0) THEN
       WRITE(IUNIT,10)
       RETURN
      ENDIF
C
C Define constants.
C
      ETA=DCMPLX(0.376730311E+03,0.000000000E+00)
      GAM=DCMPLX(0.000000000E+00,0.454756456E+01)
      PI=3.141592653589793
      CFDR=PI/180.
C
C Calculate sines and cosines.
C
      SNTH=DSIN(THETA)
      CSTH=DCOS(THETA)
      SNPH=DSIN(PHI)
      CSPH=DCOS(PHI)
C
C Find value of projected angle onto east-west plane wrt vertical.
C Determine sign factor.
C
      ALPHA=DATAN2(SNTH*CSPH,CSTH)
      SIGNA=DSIGN(1.0D0,ALPHA)
      ALPHA=-DABS(ALPHA)
C
C Find value of angle away from plane.
C Determin sign factor.
C
      BETA=PI/2.-DACOS(SNTH*SNPH)
      SIGNB=DSIGN(1.0D0,BETA)
      BETA=DABS(BETA)
C
C Write error message if beta is too large (> 3 degrees)
C
      IF(BETA.GT.3.*CFDR) THEN
       WRITE(IUNIT,11)
       RETURN
      ENDIF
C
C Set intial values to zero.
C
      EFXR=0.
      EFXI=0.
      EFYER=0.
      EFYEI=0.
```

```
      EFYNR=0.
      EFYNI=0.
      EFZR=0.
      EFZI=0.
C
C Calculate real and imaginary parts of north-south and east-west
C dependencies of cartesian components of electic field.
C
      DO I=15,1,-1
       EFXR=EFXR*ALPHA+A(I)
       EFZR=EFZR*ALPHA+G(I)
      ENDDO
      DO I=12,1,-1
       EFXI=EFXI*ALPHA+B(I)
       EFZI=EFZI*ALPHA+H(I)
       EFYER=EFYER*ALPHA+C(I)
       EFYEI=EFYEI*ALPHA+D(I)
      ENDDO
      DO I=4,1,-1
       EFYNR=EFYNR*BETA+E(I)
       EFYNI=EFYNI*BETA+F(I)
      ENDDO
C
C Combine components insuring correct sign.
C
      EFX=DCMPLX(EFXR,EFXI)
      EFY=DCMPLX(SIGNA*EFYER*SIGNB*EFYNR,SIGNA*EFYEI*SIGNB*EFYNI)
      EFZ=DCMPLX(SIGNA*EFZR,SIGNA*EFZI)
C
C Return to subroutine.
C
      RETURN
C
C Format statements.
C
 10   FORMAT('0','NAVSPASUR HALF-WAVE ARROWHEAD DIPOLE (POLYNOMIAL
     + FIT)')
 11   FORMAT('0','APPROXIMATION IS NOT VALID FOR THIS ANGLE!!!!')
      END
C**********************************************************************
```

```
C+
      SUBROUTINE ELFFLD(PHIPP,THETPP,IFLAG,IUNIT,ETA,GAMMA,EFX,EFY,EFZ)
C
C *******************************************************************
C *                                                                 *
C *    SUBROUTINE ELFFLD_RCVR                                       *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         01-AUGUST-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELPOS_RCVR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELFFLD_RCVR
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                       'CURRNT',ELFFLD_RCVR
C
C  PARENT PROGRAM:  ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the cartesian components of the range
C  independent far electric field for a horizontal half-wave dipole over
C  an infinite ground screen.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate the far electric field for a specified angle.
C
C       2. RETURN
C
C
C  INPUTS
C    EXPLICIT:
C       IFLAG   - flag : 0 => first time called
C                        1 => called previously
C       IUNIT   - output FORTRAN unit number
C       PHIPP   - angle phi (radians) wrt element
C       THETAPP - angle theta (radians) wrt element
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EFX     - x component of far electric field with range and
C                 current dependence suppressed. (mks units)
C       EFY     - y component of far electric field with range and
C                 current dependence suppressed. (mks units)
C       EFZ     - z component of far electric field with range and
C                 current dependence suppressed. (mks units)
C       ETA     - intrinsic impedance of ambient medium (mks units)
C       GAMMA   - complex propagation constan of ambient medium (mks
C                 units)
C
C    IMPLICIT:  NONE
```

```
C
C  OTHER MAJOR VARIABLES:
C
C       ANGFTR          - primary angle dependent term
C       CIM             - complex i
C       EFP             - range independent phi component of the far
C                         electric field
C       EFT             - range independent theta component of the far
C                         electric field
C       IMGFTR          - image dipole factor
C       NDRFTR          - nondirectional factor
C       PHIFTR          - phi component factor
C       THTFTR          - theta component factor
C
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,GAMMA,EFT,EFP,EFX,EFY,EFZ
      COMPLEX*16 CIM,EXPNEG,EXPPOS,NDRFTR,IMGFTR
C
C  Define constants.
C
      DATA EO,UO/8.8541879357607759D-12,1.2566370614355172D-6/
      DATA PI/3.141592653589793/
      PI2=PI/2.
      TWOPI=2.*PI
      CIM=DCMPLX(0.,1.)
      CFIM=.3048/12.
      CFDR=PI/180.
      OMEGA=2.*PI*216.980*10.**6
      GAMMA=OMEGA*DSQRT(UO*EO)*CIM
      ETA=DSQRT(UO/EO)
C
C Do coordinate rotation to place dipole in norht-south direction.
C
      PHI=PHIPP-PI2
      THETA=THETPP
C
C Calculate sines and cosines.
C
      CSPHI=DCOS(PHI)
      SNPHI=DSIN(PHI)
      CSTHET=DCOS(THETA)
      SNTHET=DSIN(THETA)
C
C Adjust for precision errors.
C
      IF(DABS(CSPHI).LT.1.D-15) CSPHI=0.
      IF(DABS(SNPHI).LT.1.D-15) SNPHI=0.
      IF(DABS(CSTHET).LT.1.D-15) CSTHET=0.
      IF(DABS(SNTHET).LT.1.D-15) SNTHET=0.
C
C Calculate angle dependent term.
C
      ANGFTR=SNTHET*CSPHI
```

```
C
C Calculate nondirectional factors.
C
      EXPNEG=CDEXP(-CIM*PI2*ANGFTR)
      EXPPOS=CDEXP(CIM*PI2*ANGFTR)
      SQRFTR=DSQRT(1.-ANGFTR*ANGFTR)
      IF(SQRFTR.NE.0.) THEN
        NDRFTR=(EXPNEG+EXPPOS)/SQRFTR
       ELSE
        NDRFTR=DCMPLX(0.,0.)
      ENDIF
C
C Calculate directional component factors.
C
      IF(SQRFTR.NE.0.) THEN
        THTFTR=-CSPHI*CSTHET/SQRFTR
        PHIFTR=SNPHI/SQRFTR
       ELSE
        THTFTR=0.
        PHIFTR=0.
      ENDIF
C
C Calculate image factor.
C
      IMGFTR=CDEXP(-CIM*3.*PI2*CSTHET)
C
C Calculate far electric field components.
C
      EFT=CIM*ETA/2./TWOPI*NDRFTR*THTFTR*(1.-IMGFTR)
      EFP=CIM*ETA/2./TWOPI*NDRFTR*PHIFTR*(1.-IMGFTR)
C
C  Convert spherical components to cartesian components.
C
      EFX=EFT*CSTHET*CSPHI-EFP*SNPHI
      EFY=EFT*CSTHET*SNPHI+EFP*CSPHI
      EFZ=-EFT*SNTHET
C
      RETURN
      END
C***********************************************************************
```

```
C+
      SUBROUTINE ELFFLD(PHI,THETA,IFLAG,IUNIT,ETA,GAM,EFX,EFY,EFZ)
C
C ***********************************************************************
C *                                                                     *
C *     SUBROUTINE ELFFLD_TRNS                                          *
C *                                                                     *
C ***********************************************************************
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         31-MARCH-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELFFLD_TRNS.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:
C                       GFFLD  - calculates the far electric field for
C                                a specified angle produced by all of
C                                the test dipole modes which model the
C                                ground screen current.
C                       NTELFF - calculates the range independent far
C                                electric field of an isolated NAVSPASUR
C                                arrowhead dipole with unit terminal
C                                current.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ELFFLD_TRNS
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                      'CURRNT',ELFFLD_TRNS
C
C  PARENT PROGRAM:   ARRAY_SBR.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the range independent far electric
C  field for a specified direction of a NAVSPASUR arrowhead dipole with
C  unit terminal current placed over a finite thin wire-grid ground
C  screen.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  IF
C                subroutine is called for the first time
C
C                    THEN
C
C                        READ
C                            - input values from THINWIRE.OUT.
C
C                        RETURN
C
C           ENDIF
C
C       2.  CALL
C                 - isolated NAVSPASUR transmitter element contribution
C                   to electric field.
C
C       3.  CALL
C                 - ground screen contribution to electic field.
C
```

```
C       4.  Calculate cartesian components of total electric field.
C
C       5.  RETURN
C
C  INPUTS
C    EXPLICIT:
C       IFLAG    - flag : 0 => first time called
C                                 1 => called previouly
C       IUNIT    - ouput unit number
C       PHI      - standard spherical coordinates phi (radians)
C       THETA    - standard spherical coordinates theta (radians)
C
C       FOR004 - THINWIRE.OUT
C                CI(K)    - imaginary part of current amplitude for test
C                           dipole mode K
C                CR(K)    - real part of current amplitude for test dipole
C                           mode K
C                D(J)     - length of segment J
C                ETA      - intrinsic impedance of ambient medium
C                GAM      - complex propagation constant of ambient medium
C                IA(J)    - 1st endpoint of segment J
C                IB(J)    - 2nd endpoint of segment J
C                INM      - maximum allowed number of segments
C                INS      - north-south wire indicator (0 : no N-S wires)
C                I1(K)    - 1st endpoint of test dipole K
C                I2(K)    - terminal point of test dipole K
C                I3(K)    - 2nd endpoint of test dipole K
C                MD(J,L)  - list of test dipoles sharing segment J
C                N        - total # of test dipole modes
C                ND(J)    - total # of test dipoles sharing segment J
C                NP       - # of points (segment endpoints)
C                NSYM     - degrees of symmetry of ground screen
C                NW       - # of east-west (parallel to element plane)
C                           ground wires
C                N1       - minimum # of test dipoles for which a current
C                           value must be calculated due to the 4-fold
C                           symmetry of at least part of the test dipole
C                           distribution (N2/4)
C                N2       - # of test dipoles which are part of the 4-fold
C                           symmetry
C                XPT(I)   - x coordinate (meters) of point I wrt element
C                           terminal
C                YPT(I)   - y coordinate (meters) of point I wrt element
C                           terminal
C                ZPT(I)   - z coordinate (meters) of point I wrt element
C                           terminal
C
C    IMPLICIT:   NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EFX      - x component of far electric field (mks units) from
C                  element with range and current dependence suppressed
C       EFY      - y component of far electric field (mks units) from
C                  element with range and current dependence suppressed
C       EFZ      - z component of far electric field (mks units) from
C                  element with range and current dependence suppressed
C       ETA      - intrinsic impedance of ambient medium (mks units)
C       GAM      - complex propagation constant of ambient medium (mks
C                  units)
```

```
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C       EFP      - phi component of far electric field from element with
C                  range and current dependence suppressed.
C       EFPD     - phi component of far electric field for isolated
C                  dipole element with range and current dependence
C                  suppressed.
C       EFPS     - phi components of far electric field scattered from
C                  ground screen with range and current dependence
C                  suppressed.
C       EFT      - theta component of far electric field from element
C                  with range and current dependence suppressed.
C       EFTD     - theta component of far electric field for isolated
C                  dipole element with range and current dependence
C                  suppressed.
C       EFTS     - theta components of far electric field scattered from
C                  ground screen with range and current dependence
C                  suppressed.
C       CGD(J)   - cosh(GAM*D(J))
C       SGD(J)   - sinh(GAM*D(J))
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 EFT,EFTD,EFTS,EFP,EFPD,EFPS,EFX,EFY,EFZ
      COMPLEX*16 ETA,GAM
      COMPLEX*16 CGD(1587),SGD(1587)
      DIMENSION  XPT(832),YPT(832),ZPT(832),D(1587)
      DIMENSION  IA(1587),IB(1587),I1(2404),I2(2404),I3(2404)
      DIMENSION  MD(1587,4),ND(1587)
      DIMENSION  CR(2404),CI(2404)
      INM=1587
C
C Execute first section only if this is the first time the subroutine is
C called.
C
      IF(IFLAG.EQ.0) THEN
C
C Write element name.
C
       WRITE(IUNIT,11)
C
C Read input values from THINWIRE.OUT.
C
       READ(4,10) ETA
       READ(4,10) GAM
       READ(4,*) NSYM
       READ(4,*) NW
       READ(4,*) INS
       READ(4,*) NP
       READ(4,*) NM
       READ(4,*) N
       READ(4,*) N1
       READ(4,*) N2
```

```
C
      DO I=1,NP
       READ(4,*) XPT(I),YPT(I),ZPT(I)
      ENDDO
C
      DO J=1,NM
       READ(4,*) IA(J),IB(J),D(J)
      ENDDO
C
      DO J=1,NM
       READ(4,*) ND(J),MD(J,1),MD(J,2),MD(J,3),MD(J,4)
      ENDDO
C
      DO K=1,N
       READ(4,*) I1(K),I2(K),I3(K)
      ENDDO
C
      DO K=1,N1
       READ(4,*) CR(K),CI(K)
      ENDDO
C
      DO J=1,NM
       CGD(J)=(CDEXP(GAM*D(J))+CDEXP(-GAM*D(J)))/2.
       SGD(J)=(CDEXP(GAM*D(J))-CDEXP(-GAM*D(J)))/2.
      ENDDO
C
C Return to subroutine.
C
      RETURN
C
     ENDIF
C
C Calculate isolated element contribution to electric field.
C
         CALL NTELFF(PHI,THETA,ETA,EFTD,EFPD)
C
C Calculate ground screen contribution to electric field.
C
         CALL GFFLD(PHI,THETA,IA,IB,INM,I1,I2,I3,XPT,YPT,ZPT,MD,N,
     +     N1,N2,ND,NM,NW,NSYM,INS,ETA,GAM,D,CGD,SGD,CR,CI,
     +     EFTS,EFPS)
C
C Calculate total electric field for element.
C
         EFT=EFTD+EFTS
         EFP=EFPD+EFPS
C
C Calculate sines and cosines of angles.
C
         CSPHI=DCOS(PHI)
         SNPHI=DSIN(PHI)
         CSTHET=DCOS(THETA)
         SNTHET=DSIN(THETA)
C
C Calculate cartesian components of electric field.
C
         EFX=EFT*CSTHET*CSPHI-EFP*SNPHI
         EFY=EFT*CSTHET*SNPHI+EFP*CSPHI
         EFZ=-EFT*SNTHET
C
```

```
C Format statements.
C
 10   FORMAT(1X,2E15.8)
 11   FORMAT('0','NAVSPASUR HALF-WAVE ARROWHEAD DIPOLE')
      RETURN
      END
C*********************************************************************
C+
      SUBROUTINE GFFLD(PHI,THETA,IA,IB,INM,I1,I2,I3,X,Y,Z,MD,N,N1,N2,
     2ND,NM,NW,NSYM,INS,ETA,GAM,D,CGD,SGD,CR,CI,ETTS,EPPS)
C
C ********************************************************************
C *                                                                  *
C *    SUBROUTINE GFFLD                                              *
C *                                                                  *
C ********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
C                            Information Service, Springfield, VA 22151,
C                            NASA Contract Report CR-2399, June 1974
C  DATE:        JUNE-1974
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.ARRAY]ELFFLD_TRNS.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C
C  SUBROUTINES CALLED:  GFF - calculates range independent far electric
C                             field for a single sinusoidal electric
C                             monopole with unit current at either
C                             endpoint.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN ARRAY
C
C  LINK/LOAD INSTRUCTIONS(TKB):  $ LINK ARRAY
C
C  PARENT PROGRAM:      ELFFLD_TRNS.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the range independent far electric
C  field for a specified angle produced by a number of test dipole
C  modes.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. DO
C               for each segment.
C
C                   CALL
C                       - electric field for coinciding test monopole
C                         for both possible orientations.
C
C                   DO
C                       for each test monopole coinciding with segment.
C
C                          Determine orientation of test monopole.
C
```

```
C                          Add test monopole's contribution to
C                          correspondingtest dipole mode's electric
C                          field.
C
C                   ENDDO
C
C           ENDDO
C
C       2.  DO
C               for each test dipole mode.
C
C                   Add its contribution to the total electric field.
C
C           ENDDO
C
C       3.  RETURN
C
C  INPUTS
C    EXPLICIT:
C       CGD      - cosh(GAM*D)
C       CI(K)    - imaginary part of current amplitude for test dipole
C                  mode K (amperes)
C       CR(K)    - real part of current amplitude for test dipole mode K
C                  (amperes)
C       D(J)     - length of segment J (meters)
C       ETA      - intrinsic impedance of ambient medium
C       GAM      - complex propagation constant of ambient medium
C       IA(J)    - 1st endpoint of segment J
C       IB(J)    - 2nd endpoint of segment J
C       INM      - maximum allowed number of segments
C       INS      - north-south wire indicator (0 : no N-S wires)
C       I1(K)    - 1st endpoint of test dipole K
C       I2(K)    - terminal point of test dipole K
C       I3(K)    - 2nd endpoint of test dipole K
C       MD(J,L)  - list of test dipoles sharing segment J
C       ND(J)    - total # of test dipoles sharing segment J
C       N        - total # of test dipole modes
C       NM       - number of segments
C       NSYM     - degrees of symmetry of ground screen
C       NW       - # of east-west (parallel to element plane) ground
C                  wires
C       N1       - minimum # of test dipoles for which a current value
C                  must be calculated due to the 4-fold symmetry of least
C                  part of the test dipole distribution (N2/4)
C       N2       - # of test dipoles which are part of the 4-fold
C                  symmetry
C       PHI      - angle phi (radians)
C       SGD      - sinh(GAM*D)
C       THETA    - angle theta (radians)
C       X(I)     - x coordinate of point I (meters) wrt element terminal
C       Y(I)     - y coordinate of point I (meters) wrt element terminal
C       Z(I)     - z coordinate of point I (meters) wrt element terminal
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EPPS     - phi component of far electric field with range
C                  dependence suppressed (mks units)
C       ETTS     - theta component of far electric field with range
```

```
C                 dependence suppressed (mks units)
C
C   IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C       EPP(I)  - phi component of range dependence suppressed far
C                 electric field of test dipole I
C       ETT(I)  - theta component of range dependence suppressed far
C                 electric field of test dipole I
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C  MODIFIED:
C       DR. STEVEN L. BERG, 03/15/88 -
C               Conformed to ELFFLD_TRNS's and THINWIRE's requirements.
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,GAM
      COMPLEX*16 CJ,ET1,EP1,ET2,EP2
      COMPLEX*16 ETTS,EPPS
      COMPLEX*16 ETT(2404),EPP(2404)
      COMPLEX*16 CGD(1),SGD(1)
      DIMENSION CR(1),CI(1)
      DIMENSION IA(1),IB(1),I1(1),I2(1),I3(1),X(1),Y(1),Z(1),D(1),ND(1)
      DIMENSION MD(INM,4)
C
C Define PI.
C
      DATA PI/3.141592653589793/
C
C Calculate sines and cosines.
C
      CSPHI=DCOS(PHI)
      SNPHI=DSIN(PHI)
      CSTHET=DCOS(THETA)
      SNTHET=DSIN(THETA)
C
C Adjust for precision errors.
C
      IF(DABS(CSPHI).LT.1.D-15) CSPHI=0.
      IF(DABS(SNPHI).LT.1.D-15) SNPHI=0.
      IF(DABS(CSTHET).LT.1.D-15) CSTHET=0.
      IF(DABS(SNTHET).LT.1.D-15) SNTHET=0.
C
C Zero initial electric fields from all test dipoles modes.
C
      DO 130 I=1,N
       ETT(I)=DCMPLX(.0,.0)
       EPP(I)=DCMPLX(.0,.0)
 130  CONTINUE
C
C Calculate electric field from test dipoles coinciding with all
C segments.
C
      DO 140 K=1,NM
       KA=IA(K)
       KB=IB(K)
```

```
C
C  Calculate electric field from test monopole coinciding with segment K
C  for both orientations.
C
        CALL GFF(X(KA),Y(KA),Z(KA),X(KB),Y(KB),Z(KB),D(K),
      2  CGD(K),SGD(K),CSTHET,SNTHET,CSPHI,SNPHI,GAM,ETA,ET1,ET2,EP1,EP2)
C
        NDK=ND(K)
C
C  For each test monopole (test dipole arm) coinciding with segment K,
C  determine the monopole's orientation wrt segment K and add its
C  contribution to its corresponding test dipole mode's electric field.
C
        DO 140 II=1,NDK
         I=MD(K,II)
         FI=1.
         IF(KB.EQ.I2(I)) GO TO 136
         IF(KB.EQ.I1(I)) FI=-1.
         EPP(I)=EPP(I)+FI*EP1
         ETT(I)=ETT(I)+FI*ET1
         GO TO 140
 136     IF(KA.EQ.I3(I)) FI=-1.
         EPP(I)=EPP(I)+FI*EP2
         ETT(I)=ETT(I)+FI*ET2
 140    CONTINUE
C
C Zero electric field produced by all test dipole modes.
C
        EPPS=(.0,.0)
        ETTS=(.0,.0)
C
C Add electric field produced by all test dipole modes.
C
        DO 260 I=1,N
         IF(NSYM.NE.1) GO TO 240
         CJ=DCMPLX(CR(I),CI(I))
         GO TO 250
 240     CONTINUE
         I4=I
         PM=1.0
         IF(I4.GT.3*N1 .AND. I4.LE.4*N1 .AND. INS.NE.0) PM=-1.0
         IF(I4.GT.3*N1 .AND. I4.LE.4*N1) I4=I4-3*N1
         IF(I4.GT.2*N1 .AND. I4.LE.4*N1) I4=I4-2*N1
         IF(I4.GT.N1 .AND. I4.LE.4*N1 .AND. INS.NE.0) PM=-1.0
         IF(I4.GT.N1 .AND. I4.LE.4*N1) I4=I4-N1
         IF(I4.GT.N2) I4=I4-(N-N2)
         IF(I4.GT.4*N1) I4=I4-3*N1
         CJ=PM*DCMPLX(CR(I4),CI(I4))
 250     CONTINUE
         ETTS=ETTS+CJ*ETT(I)
         EPPS=EPPS+CJ*EPP(I)
 260    CONTINUE
C
C  Return to subroutine
C
        RETURN
        END
C**********************************************************************
C+
        SUBROUTINE GFF(XA,YA,ZA,XB,YB,ZB,D,CGD,SGD,CTH,STH,
```

```
     2CPH,SPH,GAM,ETA,ET1,ET2,EP1,EP2)
C
C *******************************************************************
C *                                                                 *
C *     SUBROUTINE GFF                                              *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:       J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C                REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                             THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                             CONDUCTING MEDIUM", National Technical
C                             Information Service, Springfield, VA 22151,
C                             NASA Contract Report CR-2399, June 1974
C  DATE:         JUNE-1974
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELFFLD_TRNS.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:          $ FORTRAN ELFFLD_TRNS
C
C  LINK/LOAD INSTRUCTIONS(TKB):   $ LINK 'MAIN', ARRAY_SBR, 'ELPOS',
C                                        'CURRNT',ELFFLD_TRNS
C
C  PARENT PROGRAM:   GFFLD.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine uses the equations from reference 1 to calculate
C  the far-zone field of a sinusoidal electric test monopole for two
C  cases, corresponding to unit current at endpoint A and endpoint B.
C  The test monopole has endpoints (XA,YA,ZA) and (XB,YB,ZB).  The range
C  dependence is suppressed.  The far field vanishes in the endfire
C  direction.
C
C  Reference 1:
C       Richmond, J.H., "Radiation and scattering by thin-wire
C       structures in the complex frequency domain," Report 2902-10,
C       July, 1973, The Ohio State University ElectroScience Laboratory,
C       Department of Electrical Engineering: prepared under Grant NGL
C       36-008-138 for National Aeronautics and Space Administration,
C       Langley Research Center. (Available as NASA CR-2396, 1974)
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate electric field components for test monopole.
C
C       2. RETURN
C
C
C  INPUTS
C    EXPLICIT:
C       CGD      - cosh(GAM*D)
C       CPH      - cos(phi)
C       CTH      - cos(theta)
C       D        - length of test monopole (meters)
C       ETA      - intrinsic impedance of ambient medium (mks units)
C       GAM      - complex propagation constant of ambient medium (mks
```

```
C                 units)
C       SGD     - sinh(GAM*D)
C       SPH     - sin(phi)
C       STH     - sin(theta)
C       XA      - x component of 1st endpoint of test monopole (meters)
C       YA      - y component of 1st endpoint of test monopole (meters)
C       ZA      - z component of 1st endpoint of test monopole (meters)
C       XB      - x component of 2nd endpoint of test monopole (meters)
C       YB      - y component of 2nd endpoint of test monopole (meters)
C       ZB      - z component of 2nd endpoint of test monopole (meters)
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EP1     - phi component of far electric field with unit current
C                 at (XA,YA,ZA) (mks units)
C       EP2     - phi component of far electric field with unit current
C                 at (XB,YB,ZB) (mks units)
C       ET1     - theta component of far electric field with unit
C                 current at (XA,YA,ZA) (mks units)
C       ET2     - theta component of far electric field with unit
C                 current at (XB,YB,ZB) (mks units)
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C       ESA     - nondirectional factor of far electric field for case A
C                 (unit current at endpoint (XA,YA,ZA)).
C       ESB     - nondirectional factor of far electric field for case B
C                 (unit current at endpoint (XB,YB,ZB)).
C       P       - phi component factor of far electric field
C       T       - theta component factor of far electric field
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C       DR. STEVEN L. BERG, 03/05/88 -
C               Header and comments have been added, variables have been
C               changed to double precision, and one minor change was
C               made as noted below in a comment.
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 GAM,ETA
      COMPLEX*16 ET1,ET2,EP1,EP2,EGFA,EGFB,EGGD,CST
      COMPLEX*16 ESA,ESB,CGD,SGD
C
C Define constants.
C
      DATA PI/3.141592653589793/
      FP=4.*PI
C
C Calculate cartesian components of test monopole.
C
      XAB=XB-XA
```

```
      YAB=YB-YA
      ZAB=ZB-ZA
C
C Calculate direction cosines of test monopole vrt cartesian axes.
C
      CA=XAB/D
      CB=YAB/D
      CG=ZAB/D
C
C Calculate values of far field variables.
C
      G=(CA*CPH+CB*SPH)*STH+CG*CTH
C
C Added to avoid underflow.
C
      IF(DABS(G).LT.1.D-15) G=0.
      GK=1.-G*G
C
C Zero fields.
C
      ET1=(.0,.0)
      ET2=(.0,.0)
      EP1=(.0,.0)
      EP2=(.0,.0)
C
C Changed (GK.LT..001) to (GK.EQ.0.0).
C IF(GK.EQ.0.) the test monopole is in the endfire position.
C
      IF(GK.EQ.0.0) GO TO 200
C
C Calculate values of far field variables for cases A and B.
C
      FA=(XA*CPH+YA*SPH)*STH+ZA*CTH
      FB=(XB*CPH+YB*SPH)*STH+ZB*CTH
      EGFA=CDEXP(GAM*FA)
      EGFB=CDEXP(GAM*FB)
      EGGD=CDEXP(GAM*G*D)
      CST=ETA/(GK*SGD*FP)
      ESA=CST*EGFA*(EGGD-G*SGD-CGD)
      ESB=CST*EGFB*(1./EGGD+G*SGD-CGD)
C
C Calculate theta and phi component factors.
C
      T=(CA*CPH+CB*SPH)*CTH-CG*STH
      P=-CA*SPH+CB*CPH
C
C Calculate electric field components.
C
      ET1=T*ESA
      ET2=T*ESB
      EP1=P*ESA
      EP2=P*ESB
  200 CONTINUE
C
C Return to subroutine.
C
      RETURN
      END
C**********************************************************************
C+
```

```
      SUBROUTINE NTELFF(PHI,THETA,ETA,EFT,EFP)
C
C ********************************************************************
C *                                                                  *
C *    SUBROUTINE NTELFF                                             *
C *                                                                  *
C ********************************************************************
C
C
C  AUTHOR:       DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:         31-MARCH-1988
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.ARRAY]ELFFLD_TRNS.FOR
C
C  CALLING ROUTINE:     @MV7770::SPACE:[BERG.ARRAY]GEN_ARRAY.COM
C
C  SUBROUTINES CALLED:  NONE
C
C  COMPILE INSTRUCTIONS:          $ FORTRAN ELFFLD_TRNS
C
C  LINK/LOAD INSTRUCTIONS(TKB):   $ LINK 'MAIN',ARRAY_SBR,'ELPOS',
C                                        'CURRNT',ELFFLD_TRNS
C
C  PARENT PROGRAM:   ELFFLD_TRNS.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the far electric field of an isolated
C  NAVSPASUR arrowhead dipole with unit terminal current for a specified
C  angle.  The range dependence (exp(-ikr)/r) is suppressed.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate the far electric field for a specified angle.
C
C       2. RETURN
C
C  INPUTS
C    EXPLICIT:
C       ETA      - intrinsic impedance of ambient medium (mks units)
C       PHI      - angle phi (radians) wrt element
C       THETA    - angle theta (radians) wrt element
C
C    IMPLICIT:  NONE
C
C  OUTPUTS
C    EXPLICIT:
C       EFP      - range independent phi component of the far electric
C                  field (mks units)
C       EFT      - range independent theta component of the far electric
C                  field (mks units)
C
C    IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C       A        - primary angle dependent term of one monopole of dipole
C                  (monopole A)
C       B        - primary angle dependent term of other monopole of
C                  dipole (monopole B)
C       CEA      - nondirectional factor of monopole A
```

```
C       CEB     - nondirectional factor of monopole B
C       CIM     - complex i
C       PHA     - phi component factor of monopole A
C       PHB     - phi component factor of monopole B
C       THA     - theta component factor of monopole A
C       THB     - theta component factor of monopole B
C
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ETA,EFT,EFP
      COMPLEX*16 CIM,CEXPA,CEXPB,CEA,CEB
C
C Define constants.
C
      DATA PI/3.141592653589793/
      CFDR=PI/180.
      P2=PI/2.
      TP=2.*PI
      CIM=DCMPLX(.0,1.)
C
C Calculate sines and cosines.
C
      CSPHI=DCOS(PHI)
      SNPHI=DSIN(PHI)
      CSTHET=DCOS(THETA)
      SNTHET=DSIN(THETA)
C
C Adjust for precision errors.
C
      IF(DABS(CSPHI).LT.1.D-15) CSPHI=0.
      IF(DABS(SNPHI).LT.1.D-15) SNPHI=0.
      IF(DABS(CSTHET).LT.1.D-15) CSTHET=0.
      IF(DABS(SNTHET).LT.1.D-15) SNTHET=0.
C
C Calculate angle dependent terms.
C  55 degrees is the droop angle (angle below horizontal) of the
C  NAVSPASUR arrowhead dipole.
C
      CS55=DCOS(CFDR*55.)
      SN55=DSIN(CFDR*55.)
      AB=SN55*CSTHET
      BA=CS55*SNTHET*CSPHI
      A=AB+BA
      IF(A.GT.1.0) A=1.0
      IF(A.LT.-1.0) A=-1.0
      B=AB-BA
      IF(B.GT.1.0) B=1.0
      IF(B.LT.-1.0) B=-1.0
C
C Calculate nondirectional factors.
C
      CEXPA=CDEXP(-CIM*P2*A)
      CEXPB=CDEXP(-CIM*P2*B)
      SQA=DSQRT(1.-A*A)
```

```
      SQB=DSQRT(1.-B*B)
      IF(SQA.NE.0.) THEN
        CEA=(CEXPA+(CIM*A))/SQA
       ELSE
        CEA=DCMPLX(0.,0.)
      ENDIF
      IF(SQB.NE.0.) THEN
        CEB=(CEXPB+(CIM*B))/SQB
       ELSE
        CEB=DCMPLX(0.,0.)
      ENDIF
C
C Calculate directional component factors.
C
      THPH=SN55*SNTHET
      PHTH=CS55*CSTHET*CSPHI
      PHC=CS55*SNPHI
      IF(SQA.NE.0.) THEN
        THA=(THPH-PHTH)/SQA
        PHA=PHC/SQA
       ELSE
        THA=0.
        PHA=0.
      ENDIF
      IF(SQB.NE.0.) THEN
        THB=-(THPH+PHTH)/SQB
        PHB=PHC/SQB
       ELSE
        THB=0.
        PHB=0.
      ENDIF
C
C Calculate far electric field components.
C
      EFT=CIM*ETA/2./TP*(CEA*THA+CEB*THB)
      EFP=CIM*ETA/2./TP*(CEA*PHA+CEB*PHB)
C
      RETURN
      END
C***********************************************************************
```

```
C+
      PROGRAM THINWIRE
C
C *******************************************************************
C *                                                                 *
C *    PROGRAM THINWIRE                                             *
C *                                                                 *
C *******************************************************************
C
C  AUTHOR:      DR. STEVEN L. BERG, INTERFEROMETRICS INC.
C  DATE:        08-FEBRUARY-1988
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  GSCRD  - determines endpoints associated with
C                                wire grid segments and their
C                                coordinates for a NAVSPASUR transmitter
C                                ground screen
C                       SORT   - creates a set of test dipole modes for
C                                a thin-wire structure
C                       SORT4  - creates a set of test dipole modes
C                                specifically for a NAVSPASUR
C                                transmitter ground screen with only
C                                east-west wires and 4-fold symmetry
C                       SGANT  - calculates the mutual impedance matrix
C                                for the test dipole modes
C                       CROUT  - (LND=.FALSE.) decomposes a square
C                                matrix [A] into [D]
C                                (LND=.TRUE.) uses decomposed square
C                                matrix [D] to obtain column matrix [B]
C                                from the matrix equation [A][B]=[C],
C                                where [A] and [C] are known and [C] is
C                                a column matrix.
C                       VLTMTX - calculates an excitation voltage matrix
C                                for the electric field from a NAVSPASUR
C                                transmitter element incident on a
C                                thin-wire structure.
C
C  COMPILE INSTRUCTIONS:         $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PROGRAM DESCRIPTION:
C       This program uses the piecewise Galerkin method of moments to
C  calculate the complex current distribution over a wire-grid ground
C  screen of selected size which lies below a NAVSPASUR transmitter
C  element.  A cartesian coordinate system is used, with the positive
C  x, y, and z axes corresponding to the directions west, south, and up,
C  respectively.
C
C  REFERENCES :  1. J. H. Richmond, "COMPUTER PROGRAM FOR THIN-WIRE
C                   STRUCTURES IN A HOMOGENEOUS CONDUCTING MEDIUM"
C                   National Technical Information Service, Springfield,
C                   VA 22151, NASA Contract Report CR-2399, June 1974
C                2. G. A. Thiele, WIRE GRID BODY PROGRAM from "WIRE
C                   ANTENNAS", Chapter 2 in Computer Techniques for
C                   Electromagnetics, Pergamon Press, New York, 1973
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
```

```
C       1. READ
C                -degrees of symmetry
C                -north-south wire indicator
C                -wire radius
C                -wire conductivity
C                -ambient medium relative electric permittivity
C                -ambient medium conductivity
C                -# of E-W wires
C                -# of segments/wire
C                -spacing between wires
C                -height of dipole vertex above ground screen
C                -wire length
C                -transmission frequency
C
C       2. CALL
C                - endpoints associated with E-W segments and their
C                  coordinates for a NAVSPASUR transmitter ground screen.
C
C       3. CALL
C                - a set of test dipole modes for the ground screen.
C
C       4. CALL
C                - the mutual impedance matrix for the test dipole modes.
C
C       5. CALL
C                - the decomposed mutual impedance matrix.
C
C       6. CALL
C                - the excitation voltage matrix for the electic field
C                  incident on the ground screen.
C
C       7. CALL
C                - the sinusoidal current amplitude matrix of the test
C                  dipole modes.
C
C       8. END
C
C  INPUTS   EXPLICIT:
C       FOR005  - THINWIRE.INP
C                AM     - wire radius (meters)
C                CMM    - wire conductivity (mhos/meters)
C                         (zero or negative value is treated as infinite
C                         conductivity)
C                ER3    - ambient medium relative electric permittivity
C                FMC    - frequency (megahertz)
C                INS    - N-S wire indicator
C                         (0 : no N-S wires; 1 : 13 N-S wires)
C                NSW    - # of segments/E-W wire
C                NSYM   - degrees of symmetry of ground screen
C                NW     - # of E-W ground wires
C                SIG3   - ambient medium conductivity (mhos/meter)
C                SPC    - spacing between E-W wires
C                WRL    - E-W wire length
C                ZHGT   - height of element vertex above ground screen
C
C           IMPLICIT:  NONE
C
C  OUTPUTS  EXPLICIT:
C       FOR004  - THINWIRE.OUT
C                ETA      - intrinsic impedance of ambient medium
```

```
C              GAM     - complex propagation constant of ambient medium
C              INS     - N-S wire indicator
C                        (0 : no N-S wires; 1 : 13 N-S wires)
C              N       - total # of test dipole modes
C              N1      - # of test dipole modes in one quadrant of the
C                          ground screen, which have symmetric counter
C                          parts in the other three quadrants.
C              N2      - # of test dipole modes in one half of the
C                          ground screen, which have symmetric counter
C                          parts in the other half.
C              NM      - number of segments
C              NP      - number of points (segment endpoints)
C              NSYM    - degrees of symmetry of ground screen
C              NW      - # of E-W ground wires
C       FOR006  - THINWIRE.DAT
C              AM      - wire radius (meters)
C              CMM     - wire conductivity (mhos/meters)
C                          (zero or negative value is treated as infinite
C                          conductivity)
C              ER3     - ambient medium relative electric permittivity
C              FMC     - frequency (megahertz)
C              IA(J)   - 1st endpoint of segment J
C              IB(J)   - 2nd endpoint of segment J
C              MAX     - maximum value of ND(J)
C              MIN     - minimum value of ND(J)
C              N       - total # of test dipole modes
C              NSW     - # of segments/E-W wire
C              NW      - # of E-W ground wires
C              SIG3    - ambient medium conductivity (mhos/meter)
C              SPC     - spacing between E-W wires
C              WRL     - E-W wire length
C              ZHGT    - height of element vertex above ground screen
C
C           IMPLICIT:  NONE
C
C  OTHER MAJOR VARIABLES:
C
C   CFIM      - conversion factor : inches to meters
C   CGD(J)    - cosh (GAM*D(J))
C   CI(I,J)   - imaginary values of mutual impedance matrix for test
C               dipoles
C   CR(I,J)   - real values of mutual impedance matrix for test dipoles
C   D(J)      - length of segment J
C   E0        - electric permittivity of vacuum
C   EP3       - complex dielectric constant of ambient medium
C   FHZ       - frequency (hertz)
C   I1(I)     - 1st endpoint of test dipole I
C   I2(I)     - terminal endpoint of test dipole I
C   I3(I)     - 2nd endpoint of testdipole I
C   ICJ       - maximum allowed number of test dipole modes
C   IER       - error indicator for numerical integration
C   INM       - maximum allowed number of segments
C   JA(I)     - 1st segment of test dipole I
C   JB(I)     - 2nd segment of test dipole I
C   MD(J,K)   - list of test dipoles sharing segment J
C   ND(J)     - total number of test dipoles sharing segments J
C   NEF       - effective # of test dipoles (minimum # of test dipole
C               currents needed to be calculated)
C   NVL       - column number for excitation voltage column matrix
C   OMEGA     - angular frequency
```

```
C   SGD(J)     - sinh (GAM*D(J))
C   UO         - magnetic permeability of vacuum
C   WL         - wavelength (meters)
C   X(I)       - x coordinate of point I (meters)
C   Y(I)       - y coordinate of point I (meters)
C   Z(I)       - z coordinate of point I (meters)
C
C  SYSTEM STATE CHANGES: NONE
C
C-
C
C Declare variables.
C
C Avaiable memory space of computer will determine the limit of the
C size of ground screen for which this program may be used. Limits are
C reflected in matrices' sizes.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 EP3,ETA,GAM
      COMPLEX*16 CGD(3000),SGD(3000)
      DIMENSION CR(602,602),CI(602,602)
      DIMENSION I1(3000),I2(3000),I3(3000),JA(3000),JB(3000)
      DIMENSION D(3000),IA(3000),IB(3000),MD(3000,4),ND(3000)
      DIMENSION X(3000),Y(3000),Z(3000)
      LOGICAL LND
C
C Define constants.
C
      DATA EO,UO/8.854187935760759E-12,1.2566370614355172E-6/
      DATA PI/3.141592653589793/
      TP=2.*PI
      LND=.FALSE.
      ICJ=3000
      INM=3000
      CFIM=.3048DO/12.DO
C
C Read input data and write it to an output.
C
      READ(5,8) NSYM,INS
      READ(5,7) AM,CMM,ER3,SIG3
      WRITE(6,10)
      WRITE(6,17) AM,CMM,ER3,SIG3
C
      READ(5,8) NW,NSW
      WRITE(6,11)
      WRITE(6,18) NW,NSW
C
      READ(5,7) SPC,ZHGT,WRL
      WRITE(6,14) SPC
      WRITE(6,15) ZHGT
      WRITE(6,16) WRL
C
      READ(5,7) FMC
      WRITE(6,12) FMC
C
C Each point of the thin-wire model is assigned an integer #, 1 -> NP
C Each segment of the thin-wire model is assigned and integer #, 1 -> NM
C
C Determine the endpoints associated with east-west segments and their
C coordinates for a NAVSPASUR transmitter antenna ground screen.
```

```
C
      IF(NW.EQ.0) GO TO 51
      CALL GSCRD(NW,NSW,WRL,SPC,ZHGT,INS,NM,NP,IA,IB,X,Y,Z)
C
C Write endpoints associated with each antenna.
C
      WRITE(6,13)
      DO 40 J=1,NM
       WRITE(6,19) J,IA(J),IB(J)
 40   CONTINUE
      WRITE(6,22)
C
C Write endpoint positions in inches, then convert to meters.
C
      DO 50 I=1,NP
       WRITE(6,4) I,X(I),Y(I),Z(I)
       X(I)=X(I)*CFIM
       Y(I)=Y(I)*CFIM
       Z(I)=Z(I)*CFIM
 50   CONTINUE
C
C Calculate constants.
C
 51   FHZ=FMC*1.D6
      WL=1.DO/DSQRT(EO*UO)/FHZ
      OMEGA=TP*FHZ
      EP3=DCMPLX(ER3*EO,-SIG3/OMEGA)
      ETA=CDSQRT(UO/EP3)
      GAM=OMEGA*CDSQRT(-UO*EP3)
C
C Define a set of test dipole modes on the thin-wire ground screen.
C
      IF(NSYM.EQ.4) GO TO 200
      CALL SORT(IA,IB,I1,I2,I3,JA,JB,MD,ND,NM,NP,N,MAX,MIN,ICJ,INM)
      GO TO 250
 200  CALL SORT4(IA,IB,I1,I2,I3,JA,JB,MD,ND,NM,NP,N,N1,N2,MAX,MIN,ICJ,
     2INS,INM,NW,NSW)
C
 250  WRITE(6,5)
      WRITE(6,9) MAX,MIN,N
      WRITE(6,5)
C
C The number of test dipoles sharing a single segment (ND) must be at
C least one, otherwise the segment is disconnected and the computation
C is terminated.  An isolated wire must have at least two segments and
C three points.  Also ND cannot exceed four.  The total number of test
C dipole modes (N) must no exceed the maximum allowable value (ICJ).
C
      IF (MAX.GT.4 .OR. MIN.LT.1 .OR. N.GT.ICJ) GO TO 700
C
C Calculate the mutual impedance matrix.
C
      CALL SGANT(IA,IB,INM,ICJ,I1,I2,I3,JA,JB,MD,N,ND,NM,NP,NW,AM,NSYM,
     2INS,N1,N2,CGD,CMM,D,EP3,ETA,FHZ,GAM,SGD,X,Y,Z,CR,CI)
      NEF=N
      IF(NSYM.EQ.4) NEF=N1+N-N2
C
C If no test dipoles exist, only the far-field pattern of the radiating
C element is calculated.
C
```

```
 300  IF(N.LE.0) GO TO 700
      NVL=NEF+1
C
C Calculate the decomposed mutual impedance matrix using the Crout
C algorithm.
C
      CALL CROUT(NEF,NVL,CR,CI,LND)
C
C Calculate the excitation voltage matrix for an electric field incident
C on the thin-wire structure.
C
      CALL VLTMTX(NEF,N1,I1,I2,I3,X,Y,Z,ETA,GAM,WL,NVL,CR,CI,IER)
      IF(IER.NE.0) WRITE(6,25)
C
C Calculate the sinusoidal current amplitude for each test dipole.
C
      CALL CROUT(NEF,NVL,CR,CI,LND)
C
C Write input values for MV7770::SPACE:[BERG.ARRAY]ELFFLD_TRNS.FOR.
C
 700  WRITE(4,1001) ETA
      WRITE(4,1002) GAM
      WRITE(4,1003) NSYM
      WRITE(4,1004) NW
      WRITE(4,1005) INS
      WRITE(4,1006) NP
      WRITE(4,1007) NM
      WRITE(4,1008) N
      WRITE(4,1009) N1
      WRITE(4,1010) N2
      DO I=1,NP
       IF(I.EQ.1) THEN
         WRITE(4,1011) X(I),Y(I),Z(I),I
        ELSE
         WRITE(4,1012) X(I),Y(I),Z(I),I
       ENDIF
      ENDDO
      DO J=1,NM
       IF(J.EQ.1) THEN
        WRITE(4,1013) IA(J),IB(J),D(J),J
       ELSE
        WRITE(4,1014) IA(J),IB(J),D(J),J
       ENDIF
      ENDDO
      DO J=1,NM
       IF(J.EQ.1) THEN
        WRITE(4,1015) ND(J),MD(J,1),MD(J,2),MD(J,3),MD(J,4),J
       ELSE
        WRITE(4,1016) ND(J),MD(J,1),MD(J,2),MD(J,3),MD(J,4),J
       ENDIF
      ENDDO
      DO K=1,N
       IF(K.EQ.1) THEN
        WRITE(4,1017) I1(K),I2(K),I3(K),K
       ELSE
        WRITE(4,1018) I1(K),I2(K),I3(K),K
       ENDIF
      ENDDO
      DO K=1,NEF
       IF(K.EQ.1) THEN
```

```
        WRITE(4,1019) CR(K,NVL),CI(K,NVL),K
       ELSE
        WRITE(4,1020) CR(K,NVL),CI(K,NVL),K
       ENDIF
      ENDDO
C
C Format statements.
C
 4    FORMAT(1X,I5,2X,3F10.5)
 5    FORMAT(1H0)
 7    FORMAT(4F10.5)
 8    FORMAT(1X,3I5)
 9    FORMAT(3X,'MAX = ',I5,3X,'MIN = ',I5,3X,'N = ',I5)
 10   FORMAT(1X,'RADIUS (M)  SIGMA (MHOS/M)  EPSILON (REL)  SIGMA
     2(MHOS/M)')
 11   FORMAT(1X,'# OF WIRES  # OF SEGMENTS/WIRE')
 12   FORMAT(1X,'FREQUENCY = ',F7.3,' MEGAHERTZ')
 13   FORMAT(1X,'SEGMENT  1ST ENDPOINT  2ND ENDPOINT')
 14   FORMAT(1X,'WIRE SPACING = ',F3.1,' INCHES')
 15   FORMAT(1X,'ELEMENT IS ',F5.2,' INCHES ABOVE GROUND SCREEN')
 16   FORMAT(1X,'WIRE LENGTH = ',F7.3,' INCHES')
 17   FORMAT(1X,F10.5,6X,F10.5,5X,F10.5,6X,F10.5)
 18   FORMAT(6X,I5,15X,I5)
 19   FORMAT(3X,I5,9X,I5,9X,I5)
 20   FORMAT(1X,'PHI    THETA     EPHI        ETHETA')
 21   FORMAT(1X,I5,2X,I5,2X,F10.5,2X,F10.5)
 22   FORMAT(1X,'POINT     X',9X,'Y',9X,'Z')
 23   FORMAT(1X,'SCATTERED FAR FIELD')
 24   FORMAT(1X,'SCATTERED+ELEMENT FAR FIELD')
 25   FORMAT(1X,'PROBLEMS WITH NUMERICAL INTEGRATION')
 1001 FORMAT(1X,2E15.9,10X,'ETA')
 1002 FORMAT(1X,2E15.9,10X,'GAM')
 1003 FORMAT(1X,I5,35X,'NSYM - degrees of symmetry')
 1004 FORMAT(1X,I5,35X,'NW   - # of EW ground wires')
 1005 FORMAT(1X,I5,35X,'INS  - NS wire indicator')
 1006 FORMAT(1X,I5,35X,'NP   - # of points')
 1007 FORMAT(1X,I5,35X,'NM   - # of segments')
 1008 FORMAT(1X,I5,35X,'N    - # of test dipole modes')
 1009 FORMAT(1X,I5,35X,'N1')
 1010 FORMAT(1X,I5,35X,'N2')
 1011 FORMAT(1X,F10.5,1X,F10.5,1X,F10.5,1X,I4,2X,'XPT,YPT,ZPT -point
     +coordinates (meters)')
 1012 FORMAT(1X,F10.5,1X,F10.5,1X,F10.5,1X,I4)
 1013 FORMAT(1X,I5,1X,I5,1X,F10.5,1X,I4,12X,'IA,IB,D - endpoints,
     +length (meters)')
 1014 FORMAT(1X,I5,1X,I5,1X,F10.5,1X,I4)
 1015 FORMAT(1X,I5,2X,I5,1X,I5,1X,I5,1X,I5,1X,I4,5X,'ND(I).MD(I,J) - #
     +of modes, list')
 1016 FORMAT(1X,I5,2X,I5,1X,I5,1X,I5,1X,I5,1X,I4)
 1017 FORMAT(1X,I5,1X,I5,1X,I5,1X,I4,17X,'I1,I2,I3 - test dipole mode
     +points')
 1018 FORMAT(1X,I5,1X,I5,1X,I5,1X,I4)
 1019 FORMAT(1X,F15.8,1X,F15.8,1X,I4,4X,'CR,CI - real and imaginary
     +currents')
 1020 FORMAT(1X,F15.8,1X,F15.8,1X,I4)
C
 800  STOP
      END
C
C***********************************************************************
```

```
C
C+
      SUBROUTINE SORT(IA,IB,I1,I2,I3,JA,JB,MD,ND,NM,NP,N,MAX,MIN,ICJ,
     2INM)
C
C *********************************************************************
C *                                                                   *
C *     SUBROUTINE SORT                                               *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
C                            Information Service, Springfield, VA 22151,
C                            NASA Contract Report CR-2399, June 1974
C  DATE:        JUNE-1974
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED: NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine defines a set of test dipole modes on a
C  thin-wire structure.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  DO
C               for each endpoint.
C
C                  Determine # of test dipole modes with terminal at
C                  that endpoint.
C
C                  DO
C                       for each test dipole mode with terminal at
C                       that endpoint.
C
C                          Determine corresponding segments.
C
C                          Determine corresponding endpoints.
C
C                   ENDDO
C           ENDDO
C
C       2.  Determine total number of test dipoles.
C
C       3.  DO
C               for each segment.
C
C                  Determine total # of test dipoles sharing that
```

```
C                  segment.
C
C                  Create list of test dipoles sharing that segment.
C
C           ENDDO
C
C
C      4.   Determine maximum and minimum number of test dipoles sharing
C           any one segment.
C
C      5.   RETURN
C
C  INPUTS   EXPLICIT:
C      IA(J)    - 1st endpoint of segment J
C      IB(J)    - 2nd endpoint of segment J
C      NM       - # of segments
C      NP       - # of endpoints
C      ICJ      - maximum allowed number of test dipole modes
C      INM      - maximum allowed number of segments
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C      N        - total number of test dipole modes
C      I1(I)    - 1st endpoint of test dipole I
C      I2(I)    - terminal point of test dipole I
C      I3(I)    - 2nd endpoint of test dipole I
C      JA(I)    - 1st segment of test dipole I
C      JB(I)    - 2nd segment of test dipole I
C      MD(J,K)  - list of test dipoles sharing segment J
C      ND(J)    - total number of test dipoles sharing segment J
C      MAX      - maximum value of ND(J)
C      MIN      - minimum value of ND(J)
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C      JSP(I)   - list of segments intersecting at endpoint I
C      MOD      - # of test dipoles with terminals at endpoint I
C      NJK      - # of segments intersecting at endpoint I
C
C  SYSTEM STATE CHANGES: NONE
C
C-
C
C  MODIFIED:
C      Dr. Steven L. Berg, 11/25/88
C               - Added header and documentation.
C
      DIMENSION JSP(20)
      DIMENSION I1(1),I2(1),I3(1),JA(1),JB(1)
      DIMENSION IA(1),IB(1),ND(1),MD(INM,4)
      I=0
      DO 24 K=1,NP
       NJK=0
C
C Determine NJK and JSP.
C
       DO 20 J=1,NM
        IND=(IA(J)-K)*(IB(J)-K)
```

```
          IF (IND.NE.0)GO TO 20
          NJK=NJK+1
          JSP(NJK)=J
 20      CONTINUE
C
C Determine MOD.
C
         MOD=NJK-1
C
C If MOD = 0, then K is an endpoint of a single isolated segment.
C
         IF (MOD.LE.0)GO TO 24
C
C Set endpoints of test dipole modes.
C
         DO 22 IMD=1,MOD
          I=I+1
C
C Determine if number of modes exceeds maximum allowable.
C
          IF(I.GT.ICJ)GO TO 22
C
C Determine JA,JB,I1,I2, and I3 for each test dipole mode.
C
          IPD=IMD+1
          JAI=JSP(IMD)
          JA(I)=JAI
          JBI=JSP(IPD)
          JB(I)=JBI
          I1(I)=IA(JAI)
          IF(IA(JAI).EQ.K)I1(I)=IB(JAI)
          I2(I)=K
          I3(I)=IA(JBI)
          IF(IA(JBI).EQ.K)I3(I)=IB(JBI)
 22      CONTINUE
 24     CONTINUE
C
C Determine total number of test dipoles.
C
        N=I
C
C Set intital values of ND(J) and MD(J,K) equal to zero.
C
        DO 30 J=1,NM
         ND(J)=0
         DO 29 K=1,4
          MD(J,K)=0
 29      CONTINUE
 30     CONTINUE
        III=N
        IF(N.GT.ICJ)III=ICJ
C
C Determine ND(J) and MD(J,K).
C
        DO 40 I=1,III
         J=JA(I)
         DO 38 L=1,2
          ND(J)=ND(J)+1
          K=1
          M=0
```

```
 32     MJK=MD(J,K)
        IF(MJK.NE.0)GO TO 34
        M=1
        MD(J,K)=I
 34     K=K+1
        IF(K.GT.4)GO TO 38
        IF(M.EQ.0)GO TO 32
 38     J=JB(I)
 40   CONTINUE
C
C Determine MAX and MIN.
C
      MIN=100
      MAX=0
      DO 46 J=1,NM
       NDJ=ND(J)
       IF(NDJ.GT.MAX)MAX=NDJ
       IF(NDJ.LT.MIN)MIN=NDJ
 46   CONTINUE
C
C Return to main program.
C
      RETURN
      END
C
C***********************************************************************
C+
      SUBROUTINE SGANT(IA,IB,INM,ICJ,I1,I2,I3,JA,JB,MD,N,ND,NM,NP,NW,AM,
     2NSYM,INS,N1,N2,CGD,CMM,D,EP3,ETA,FHZ,GAM,SGD,X,Y,Z,CR,CI)
C
C **********************************************************************
C *                                                                    *
C *     SUBROUTINE SGANT                                               *
C *                                                                    *
C **********************************************************************
C
C  AUTHOR:       J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C                REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                             THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                             CONDUCTING MEDIUM", National Technical
C                             Information Service, Springfield, VA 22151,
C                             NASA Contract Report CR-2399, June 1974
C  DATE:         JUNE-1974
C  LANGUAGE:     VAX FORTRAN
C  FILE:         MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  CBES    - calculates the quantity
C                                 B01=J0(z)/J1(z) where z is complex and
C                                 J0 and J1 denote the Bessel functions
C                                 of order zero and one.
C                       GGS     - calculates the mutual impedance
C                                 between two filamentary monopoles with
C                                 sinusoidal current distribution using
C                                 either the GGMM subroutine or Simpson's
C                                 rule.
C                       GGMM    - calculates the mutual impedance
C                                 between two filamentary monopoles with
C                                 sinusoidal current distribution using
```

```
C                               closed-form expressions in terms of
C                               exponential integrals.
C
C  COMPILE INSTRUCTIONS:       $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the mutual impedances between test
C  dipoles and stores them in a square matrix.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate wire surface impedance.
C
C       2. Calculate mutual impedance between all possible monopole
C          pairs.
C
C       3. Fill the mutual impedance matrix.
C
C       4. RETURN
C
C  INPUTS   EXPLICIT:
C       IA(J)   - 1st endpoint of segment J
C       IB(J)   - 2nd endpoint of segment J
C       INM     - maximum allowed # of segments
C       ICJ     - maximum allowed # of simultaneous linear equations
C       I1(I)   - 1st endpoint of test dipole I
C       I2(I)   - terminal point of test diple I
C       I3(I)   - 2nd endpoint of test dipole I
C       JA(I)   - 1st segment of test dipole I
C       JB(I)   - 2nd segment of test dipole I
C       MD(J,K) - list of test dipoles sharing segment J
C       N       - total # of test dipoles
C       ND(J)   - total # of test dipoles sharing segment J
C       NM      - # of segments.
C       NP      - # of endpoints
C       AM      - wire radius (meters)
C       CMM     - wire conductivity (mhos/meter)
C                 (CMM.LE.O. is treated as infinite conductivity)
C       EP3     - complex dielectric constant of ambient medium
C       ETA     - intrinsic impedance of ambient medium
C       FHZ     - transmission frequency (Hertz)
C       GAM     - complex propagation constant of ambient medium
C       X(I)    - x coordinate of endpoint I
C       Y(I)    - y coordinate of endpoint I
C       Z(I)    - z coordinate of endpoint I
C
C            IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       CR(I,J) - real values of thin-wire impedance matrix
C       CI(I,J) - imaginary values of thin-wire impedance matrix
C       D(J)    - length of segment J
C       CGD(J)  - cosh (GAM*D(J))
C       SGD(J)  - sinh (GAM*D(J))
C
C            IMPLICIT:
```

```
C
C  OTHER MAJOR VARIABLES:
C       ZARG    - complex number used to determine ZS.
C       ZS      - wire surface impedance.
C
C  SYSTEM STATE CHANGES: NONE
C
C-
C
C  MODIFIED:
C       Dr. Steven L. BERG, 2/08/88
C               - Adapted to THINWIRE's requirements.
C       Dr. Steven L. BERG, 11/25/88
C               - Added header and documentation.
C
C
C Declare variables.
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 EGD,CGDS,SGDS,SGDT,Q11,Q12,GD,ZG,P11,P12,P21,P22
      COMPLEX*16 CGD(1),SGD(1)
      COMPLEX*16 B01
      COMPLEX*16 EP,ETA,GAM,EP3,ZS,ZH
      COMPLEX*16 EPSILA,CWEA,BETA,ZARG
      COMPLEX*16 P(2,2),Q(2,2)
      DIMENSION CR(602,602),CI(602,602)
      DIMENSION X(1),Y(1),Z(1),D(1),IA(1),IB(1),MD(INM,4)
      DIMENSION I1(1),I2(1),I3(1),JA(1),JB(1),ND(1)
      DATA EO,UO/8.8541879357607759E-12,1.2566370614355172E-6/
      DATA PI/3.141592653589793/
 2    FORMAT(3X,'AM = ',E10.3,3X,'DMAX = ',E10.3,3X,'DMIN = ',E10.3)
      TP=2.*PI
      EP=EP3
C
C Calculate wire surface impedance (ZS)
C
      ZS=(.0,.0)
C
C CMM.LE.0. is treated as infinite wire conductivity.
C
      IF(CMM.LE.0.) GO TO 12
      OMEGA=TP*FHZ
      EPSILA=DCMPLX(EO,-CMM*1.D6/OMEGA)
      CWEA=(.0,1.)*OMEGA*EPSILA
      BETA=OMEGA*DSQRT(UO)*CDSQRT(EPSILA-EP)
      ZARG=BETA*AM
C
C  Calculate the quantity B01=J0(Z)/J1(Z), where J0 and J1 denote the
C  Bessel functions of order zero and one.
C
      CALL CBES(ZARG,B01)
      ZS=BETA*B01/CWEA
 12   ZH=ZS/(TP*AM*GAM)
C
C  Determine segment lenths [D(J)] and minimum (DMIN) and maximum
C  (DMAX) values.
C
      DMIN=1.D30
      DMAX=.0
      DO 20 J=1,NM
```

```
      K=IA(J)
      L=IB(J)
      D(J)=DSQRT((X(K)-X(L))**2+(Y(K)-Y(L))**2+(Z(K)-Z(L))**2)
      IF(D(J).LT.DMIN)DMIN=D(J)
      IF(D(J).GT.DMAX)DMAX=D(J)
      EGD=CDEXP(GAM*D(J))
      CGD(J)=(EGD+1./EGD)/2.
      SGD(J)=(EGD-1./EGD)/2.
 20   CONTINUE
C
C  Length Test : If any segment length is less than the diameter of the
C  wire of if any segment length is too long , return to main program.
C
      IF(DMIN.LT.2.*AM) GO TO 25
C
C  Radius Test : If the wire radius is too large for convergence of is
C  the wire radius is zero, return to main program.
C
      IF(CDABS(GAM*AM).GT.0.06) GO TO 25
      IF (AM.GT.0.) GO TO 30
 25   N=0
      WRITE(6,2) AM,DMAX,DMIN
      RETURN
C
C  Fill impedance matrix.
C
C  Consider all segments (K)
C
 30   DO 200 K=1,NM
       NDK=ND(K)
       KA=IA(K)
       KB=IB(K)
       DK=D(K)
       CGDS=CGD(K)
       SGDS=SGD(K)
C
C  Consider all segments (L) interacting with segment K.
C
       DO 200 L=1,NM
        NDL=ND(L)
        LA=IA(L)
        LB=IB(L)
        DL=D(L)
        SGDT=SGD(L)
        NIL=0
C
C  Consider all test dipoles sharing segment K.
C
        DO 200 II=1,NDK
C
C  I is the test dipole sharing segment K
C
         I=MD(K,II)
         IF(NSYM.EQ.4 .AND. I.GT.N1 .AND. I.LE.4*N1) GO TO 200
         IF(NSYM.EQ.4 .AND. I.GT.N2) GO TO 200
         MM=(I-1)*N-(I*I-I)/2
         FI=1.
C
C  Determine if 2nd endpoint of K is terminal point of test dipole I.
C
```

```
        IF(KB.EQ.I2(I)) GO TO 36
C
C  Determine if 2nd endpoint of K is 1st endpoint of tes dipole I.
C
        IF(KB.EQ.I1(I))FI=-1.
C
C  IS denotes monopole half of test dipole under consideration.
C
        IS=1
        GO TO 40
C
C  Determine if 1st endpoint of K is 2nd endpoint of test dipole I.
C
 36     IF (KA.EQ.I3(I))FI=-1.
        IS=2
C
C  Consider all test dipoles sharing segment L.
C
 40      DO 200 JJ=1,NDL
C
C  J is the test dipole sharing segment L.
C
         J=MD(L,JJ)
C
C  C is a symmetric matrix, therefore only half the matrix needs to be
C  calculated.
C
         IF(NSYM.NE.1) GO TO 42
         IF(I.GT.J) GO TO 200
 42      FJ=1.
C
C  Determine if 2nd endpoint of L is terminal point of test dipole J.
C
         IF(LB.EQ.I2(J)) GO TO 46
C
C  Determine if 2nd endpoint of L is 1st endpoint of test dipole I.
C
         IF(LB.EQ.I1(J)) FJ=-1.
C
C  JS denotes monpole half of test dipole under consideration.
C
         JS=1
         GO TO 50
C
C  Determine if 1st endpoint of L is 2nd endpoint of test dipole I.
C
 46      IF(LA.EQ.I3(J))FJ=-1.
         JS=2
C
C  Determine if matrix contribution has already been calculated.
C
 50      IF(NIL.NE.0) GO TO 168
         NIL=1
         IF(K.EQ.L) GO TO 120
         IND1=(LA-KA)*(LB-KA)
         IND2=(LA-KB)*(LB-KB)
         IF(IND1.EQ.0 .OR. IND2.EQ.0) GO TO 80
C
C  Segments K and L share no points.
C
```

```
C  Calculate the monopole-monopole mutual impedances for all four
C  monopole-monpole interactions.
C
            CALL GGS(X(KA),Y(KA),Z(KA),X(KB),Y(KB),Z(KB),X(LA),Y(LA),
     2      Z(LA),X(LB),Y(LB),Z(LB),AM,DK,CGDS,SGDS,DL,SGDT,ETA,GAM,
     3      P(1,1),P(1,2),P(2,1),P(2,2))
            GO TO 168
C
C  Segments K and L share one point (they intersect).
C
 80         KG=0
            JM=KB
            JC=KA
            KF=1
C
C  Determine which K endpoint is shared with L.
C
            IND=(KB-LA)*(KB-LB)
            IF(IND.NE.0) GO TO 82
C
C  2nd K endpoint is shared with L.
C
            JC=KB
            KF=-1
            JM=KA
            KG=3
C
C  1st K endpoint is shared with L.
C
 82         LG=3
            JP=LA
            LF=-1
            IF(LB.EQ.JC)GO TO 83
C
C  1st L endpoint is share with K.
C
            JP=LB
            LF=1
            LG=0
C
C  2nd L endpoint is shared with K.
C
 83         SGN=KF*LF
            CPSI=((X(JP)-X(JC))*(X(JM)-X(JC))+(Y(JP)-Y(JC))*(Y(JM)
     2      -Y(JC))+(Z(JP)-Z(JC))*(Z(JM)-Z(JC)))/(DK*DL)
C
C  Calculate the monopole-monopole mutual impedances for all four
C  monopole-monpole interactions.
C
            CALL GGMM(.0,DK,.0,DL,AM,CGDS,SGDS,SGDT,CPSI,ETA,GAM,
     2      Q(1,1),Q(1,2),Q(2,1),Q(2,2))
            DO 98 KK=1,2
             KP=IABS(KK-KG)
             DO 98 LL=1,2
              LP=IABS(LL-LG)
              P(KP,LP)=SGN*Q(KK,LL)
 98         CONTINUE
            GO TO 168
C
C  Segment K is the same as segment L (self reaction of segment K).
```

```
C
 120       Q11=(.0,.0)
           Q12=(.0,.0)
C
C  CMM.LE.0. is treated as infinite conductivity.
C
           IF(CMM.LE.0.)GO TO 155
           GD=GAM*DK
           ZG=ZH/(SGDS**2)
           Q11=ZG*(SGDS*CGDS-GD)/2.
           Q12=ZG*(GD*CGDS-SGDS)/2.
 155       CONTINUE
C
C  Calculate the monopole-monopole mutual impedances for all four
C  monopole-monopole interactions.
C
           CALL GGMM(.0,DK,.0,DK,AM,CGDS,SGDS,SGDS,1.,
     2      ETA,GAM,P11,P12,P21,P22)
           Q11=P11+Q11
           Q12=P12+Q12
           P(1,1)=Q11
           P(1,2)=Q12
           P(2,1)=Q12
           P(2,2)=Q11
C
C  Determine relative orientation for proper impedance.
C
           IF(KA.NE.LA)GO TO 160
           GO TO 168
 160       P(1,1)=-Q12
           P(1,2)=-Q11
           P(2,1)=-Q11
           P(2,2)=-Q12
C
C  Determine mutual impedance matrix element between test dipole I and
C  test dipole J.
C
 168       IF(NSYM.NE.1) GO TO 170
           CR(I,J)=CR(I,J)+FI*FJ*DREAL(P(IS,JS))
           CI(I,J)=CI(I,J)+FI*FJ*DIMAG(P(IS,JS))
           CR(J,I)=CR(I,J)
           CI(J,I)=CI(I,J)
           GO TO 200
 170       CONTINUE
           J4=J
           PM=1.0
           IF(J4.GT.3*N1 .AND. J4.LE.4*N1 .AND. INS.NE.0) PM=-1.0
           IF(J4.GT.3*N1 .AND. J4.LE.4*N1) J4=J4-3*N1
           IF(J4.GT.2*N1 .AND. J4.LE.4*N1) J4=J4-2*N1
           IF(J4.GT.N1 .AND. J4.LE.4*N1 .AND. INS.NE.0) PM=-1.0
           IF(J4.GT.N1 .AND. J4.LE.4*N1) J4=J4-N1
           IF(J4.GT.N2) J4=J4-(N-N2)
           IF(J4.GT.4*N1) J4=J4-3*N1
           I4=I
           IF(I4.GT.4*N1) I4=I4-3*N1
           CR(I4,J4)=CR(I4,J4)+PM*FI*FJ*DREAL(P(IS,JS))
           CI(I4,J4)=CI(I4,J4)+PM*FI*FJ*DIMAG(P(IS,JS))
 200  CONTINUE
C
C  Return to main program.
```

```
C
      RETURN
      END
C
C***********************************************************************
C+
      SUBROUTINE CBES(Z,B01)
C
C ***********************************************************************
C *                                                                     *
C *    SUBROUTINE CBES                                                  *
C *                                                                     *
C ***********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
C                            Information Service, Springfield, VA 22151,
C                            NASA Contract Report CR-2399, June 1974
C  DATE:        JUNE-1974
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:    MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED: NONE
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:     THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This program calculates the quantity B01=J0(z)/J1(z) where z is
C  complex and J0 and J1 denote the Bessel functions of order zero and
C  one.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1. Calculate parts for Bessel functions of order zero (J0) and
C          one (J1).
C
C       2. Determine J0/J1
C
C       3. RETURN
C
C  INPUTS   EXPLICIT:
C       Z        - complex number
C
C           IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C       B01      - J0(Z)/J1(Z) wher J0 and J1 denote the Bessel functions
C                  of order zero and one
C
C           IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:  NONE
```

```
C
C  SYSTEM STATE CHANGES:   NONE
C
C-
C
C  MODIFIED:
C       Dr. Steven L. Berg, 11/30/88
C               - Added header
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 ARG,CC,CS,EX
      COMPLEX*16 B01,Z,TERMJ,TERMN,MZ24,JN(2)
      IF (CDABS(Z).GE.12.0) GO TO 10
      FACTOR=0.0
      TERMN=(0.,0.)
      MZ24=-0.25*Z*Z
      TERMJ=(1.0,0.0)
      DO 1 NP=1,2
      N=NP-1
      JN(NP)=TERMJ
      M=0
 2    M=M+1
      TERMJ=TERMJ*MZ24/FLOAT(M*(N+M))
      JN(NP)=JN(NP)+TERMJ
      IF(NP.NE.1) GO TO 3
      FACTOR=FACTOR+1.0/FLOAT(M)
      TERMN=TERMN+TERMJ*FACTOR
 3    ERROR=CDABS(TERMJ)
      IF(ERROR.GT.1.0E-10) GO TO 2
 1    TERMJ=0.5*Z
      B01=JN(1)/JN(2)
      RETURN
 10   Y=DIMAG(Z)
      IF(ABS(Y).GT.20.) GO TO 20
      ARG=(.0,1.)*Z
      EX=CDEXP(ARG)
      CC=EX+1./EX
      CS=(.0,-1.)*(EX-1./EX)
      B01=(CS+CC)/(CS-CC)
      RETURN
 20   B01=(.0,-1.)
      IF(Y.LT.O.) B01=(.0,1.)
      RETURN
      END
C
C**********************************************************************
C+
      SUBROUTINE GGS(XA,YA,ZA,XB,YB,ZB,X1,Y1,Z1,X2,Y2,Z2,AM,
     2DS,CGDS,SGDS,DT,SGDT,ETA,GAM,P11,P12,P21,P22)
C
C *********************************************************************
C *                                                                   *
C *     SUBROUTINE GGS                                                *
C *                                                                   *
C *********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
```

```
C                             Information Service, Springfield, VA 22151,
C                             NASA Contract Report CR-2399, June 1974
C  DATE:          JUNE-1974
C  LANGUAGE:      VAX FORTRAN
C  FILE:          MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  GGMM    - calculates the mutual impedance
C                                 between two filamentary monopoles with
C                                 sinusoidal current distribution using
C                                 closed-form expressions in terms of
C                                 exponential integrals.
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the mutual impedance between two
C  filamentary monopoles with sinusoidal current distribution using
C  either the GGMM subroutine or Simpson's rule.  The calculation is
C  determined by input parameter values. For more information consult
C  APPENDIX 5 in
C                 J. H. Richmond, "COMPUTER PROGRAM FOR
C                 THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                 CONDUCTING MEDIUM", National Technical
C                 Information Service, Springfield, VA 22151,
C                 NASA Contract Report CR-2399, June 1974.
C
C  PROGRAM ALGORITHM (PSEUDOCODE):
C
C       1.  Calculate mutual impedance.
C
C       2.  RETURN
C
C  INPUTS   EXPLICIT:
C       AM      -
C       CGDS    -
C       DS      -
C       DT      -
C       ETA     -
C       GAM     -
C       SGDS    -
C       SGDT    -
C       X1      -
C       X2      -
C       XA      -
C       XB      -
C       Y1      -
C       Y2      -
C       YA      -
C       YB      -
C       Z1      -
C       Z2      -
C       ZA      -
C       ZB      -
C
```

```
C            IMPLICIT:   NONE
C
C  OUTPUTS  EXPLICIT:
C
C       P11      -
C       P12      -
C       P21      -
C       P22      -
C
C            IMPLICIT:   NONE
C
C  OTHER MAJOR VARIABLES:
C       INT      - 0          => closed form calculation performed
C                  otherwise => Simpson's rule may be used
C
C  SYSTEM STATE CHANGES:  NONE
C
C-
C
C  MODIFIED:
C       DR. STEVEN L. BERG, 02/08/88
C                - INT was set equal to zero to always select closed form
C                  calculation.
C
C
      IMPLICIT REAL*8(A-H,O-Z)
      COMPLEX*16 CGDS,SGDS,SGDT,P11,P12,P21,P22,CST,ET1,ET2
      COMPLEX*16 ETA,GAM
      DATA PI/3.141592653589793/
C
C Set INT=0 to always have closed-form calculation.
C
      INT=0
C
C
C
      CA=(X2-X1)/DT
      CB=(Y2-Y1)/DT
      CG=(Z2-Z1)/DT
      CAS=(XB-XA)/DS
      CBS=(YB-YA)/DS
      CGS=(ZB-ZA)/DS
      CC=CA*CAS+CB*CBS+CG*CGS
      IF(ABS(CC).GT..997)GO TO 200
 20   SZ=(X1-XA)*CAS+(Y1-YA)*CBS+(Z1-ZA)*CGS
      GO TO 300
 200  SZ1=(X1-XA)*CAS+(Y1-YA)*CBS+(Z1-ZA)*CGS
      RH1=DSQRT((X1-XA-SZ1*CAS)**2+(Y1-YA-SZ1*CBS)**2
     2+(Z1-ZA-SZ1*CGS)**2)
      SZ2=SZ1+DT*CC
      RH2=DSQRT((X2-XA-SZ2*CAS)**2+(Y2-YA-SZ2*CBS)**2
     2+(Z2-ZA-SZ2*CGS)**2)
      DDD=(RH1+RH2)/2.
      IF(DDD.GT.20.*AM .AND. INT.GT.0) GO TO 20
      IF (DDD.LT.AM)DDD=AM
      CALL GGMM(.0,DS,SZ1,SZ2,DDD,CGDS,SGDS,SGDT,1.,ETA,GAM,
     2P11,P12,P21,P22)
      RETURN
 300  SS=DSQRT(1.-CC*CC)
      CAD=(CGS*CB-CBS*CG)/SS
```

```
      CBD=(CAS*CG-CGS*CA)/SS
      CGD=(CBS*CA-CAS*CB)/SS
      DK=(X1-XA)*CAD+(Y1-YA)*CBD+(Z1-ZA)*CGD
      DK=ABS(DK)
      IF(DK.LT.AM) DK=AM
      XZ=XA+SZ*CAS
      YZ=YA+SZ*CBS
      ZZ=ZA+SZ*CGS
      XP1=X1-DK*CAD
      YP1=Y1-DK*CBD
      ZP1=Z1-DK*CGD
      CAP=CBS*CGD-CGS*CBD
      CBP=CGS*CAD-CAS*CGD
      CGP=CAS*CBD-CBS*CAD
      P1=CAP*(XP1-XZ)+CBP*(YP1-YZ)+CGP*(ZP1-ZZ)
      T1=P1/SS
      S1=T1*CC-SZ
      CALL GGMM(S1,S1+DS,T1,T1+DT,DK,CGDS,SGDS,SGDT,CC,ETA,GAM,
     2P11,P12,P21,P22)
      RETURN
      END
C
C*********************************************************************
C+
      SUBROUTINE GGMM(S1,S2,T1,T2,D,CGDS,SGD1,SGD2,CPSI,ETA,GAM,
     2P11,P12,P21,P22)
C
C ********************************************************************
C *                                                                  *
C *    SUBROUTINE GGMM                                               *
C *                                                                  *
C ********************************************************************
C
C  AUTHOR:      J. H. RICHMOND, OHIO STATE UNIVERSITY ELECTROSCIENCE LAB
C               REFERENCE :  J. H. Richmond, "COMPUTER PROGRAM FOR
C                            THIN-WIRE STRUCTURES IN A HOMOGENEOUS
C                            CONDUCTING MEDIUM", National Technical
C                            Information Service, Springfield, VA 22151,
C                            NASA Contract Report CR-2399, June 1974
C  DATE:        JUNE-1974
C  LANGUAGE:    VAX FORTRAN
C  FILE:        MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  CALLING ROUTINE:     MV7770::SPACE:[BERG.THINWIRE]THINWIRE.FOR
C
C  SUBROUTINES CALLED:  EXPJ    - evaluates an exponential integral
C
C  COMPILE INSTRUCTIONS:        $ FORTRAN THINWIRE
C
C  LINK/LOAD INSTRUCTIONS(TKB): $ LINK THINWIRE
C
C  PARENT PROGRAM:      THINWIRE.FOR
C
C  PROGRAM DESCRIPTION:
C       This subroutine calculates the mutual impedance between two
C  filamentary monopoles with sinusoidal current distribution using
C  closed-form expressions in terms of exponential integrals. For more
C  information consult APPENDIX 6 in
C               J. H. Richmond, "COMPUTER PROGRAM FOR
C               THIN-WIRE STRUCTURES IN A HOMOGENEOUS
```